RICE UNIVERSITY

DESIGN OF FINITE IMPULSE RESPONSE DIGITAL FILTERS

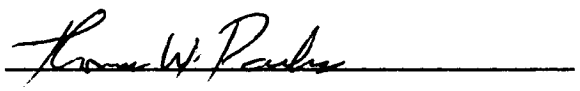
by

John D. Fisher

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Thesis Director's Signature:



Houston, Texas

May, 1973

ACKNOWLEDGEMENTS

The author gratefully acknowledges the encouragement and support of his wife Christie, and the many fruitful discussions with his thesis advisor, Dr. T. W. Parks.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

Recently, there has been interest in the design of finite length digital filters. Several reasons for this interest lie in the advantages of the finite impulse response (FIR) digital filter; namely

1. stability of the filter,
2. high speed implementation, and
3. relative insensitivity to round off in non-recursive realizations.

In the most general case, the Chebyshev approximation problem is to find the h_i, i=0,...,N such that one

$$\underset{\text{all } h_i \,\varepsilon\, \text{Real}}{\text{minimizes}} \quad \text{the } \{\max_{F \varepsilon X} |D(F) - O(F)|\}$$

where O(F) equals the frequency response of the digital filter defined as equaling the transfer function of the digital filter

$$\left(\sum_{k=0}^{N} h_k z^{-k} = G(z) \right)$$

evaluated at $e^{j\omega T} = e^{j2\pi F}$; F a normalized frequency variable $= \omega T/2\pi \in X = [0,1)$.

$$\text{(Hence } O(F) = \sum_{k=0}^{N} h_k e^{-j2\pi Fk} \text{ .)}$$

D(F) is some desired response, and $|X(F)|$ denotes the magnitude of X(F) evaulated at the frequency F.

Remark 1-1. One notes that for low pass filters with constant group delay

$$D(F) = \begin{cases} e^{-j2\pi F\gamma(N/2)} & F \in \text{Passband} \\ 0 & F \in \text{Stopband} \end{cases}$$

where γ is a linear phase factor.

Methods for solving this problem include windowing techniques, frequency sampling techniques and optimal Chebyshev approximation with linear phase.

Windowing techniques derive their name from the method in which the h_i are obtained. Since the frequency response of a digital filter is periodic, the frequency response function can be expanded in terms of a Fourier series, which in general contains an infinite number of terms. The Fourier coefficients are then truncated to form the h_i of a filter producing approximately the desired frequency response. One of the problems with the method is

at discontinuities where one usually obtains unsatisfactory performance due to the Gibbs phènomena. To avoid this difficulty, one modifies the Fourier coefficients by multiplying the coefficients by a desirable time limited function known as a window, which reduces the Gibbs effect [1].

A second method for obtaining the filter coefficient is the frequency sampling method [2]. Basically the method samples a desired frequency response at N points with the values of M points (M < N) left unspecified. The inverse discrete Fourier transform (IDFT) is then applied and the h_i of a filter determined, and a new continuous frequency response of these h_i's is found. An optimizing technique is applied to the M values left unspecified, and the filter coefficients finally achieved incorporating the newly specified value of the M points. Typical results include a maximum deviation of about -35 to -40 db for a filter of length 16 with 1 transition point for a low pass filter.

The third and most recent advancement is the fast and efficient computation of an optimal Chebyshev approximation by a FIR digital filter to a desired frequency response with linear phase [3], [4]. This interpolative technique applies the Remes Algorithm [5].

Methods one and two (windowing and frequency sampling) offer the possibility of complex approximation, but they are not optimal in the Chebyshev sense. Method 3 offers the advantage of a fast efficient procedure

(as does 2) and is optimal. However, method 3 does not offer the generality of a complex approximation since the phase is fixed to be exactly linear (thereby making the approximation problem a real approximation problem).

Remark 1-2. If one desires linear phase, then he is done, for the approximation found in [3] and [4] are optimal.

There exist, however, reasons why one would be interested in a general complex approximation to a desired response in the Chebyshev sense. Consider the cross sectional plane shown in Figure 1-1.

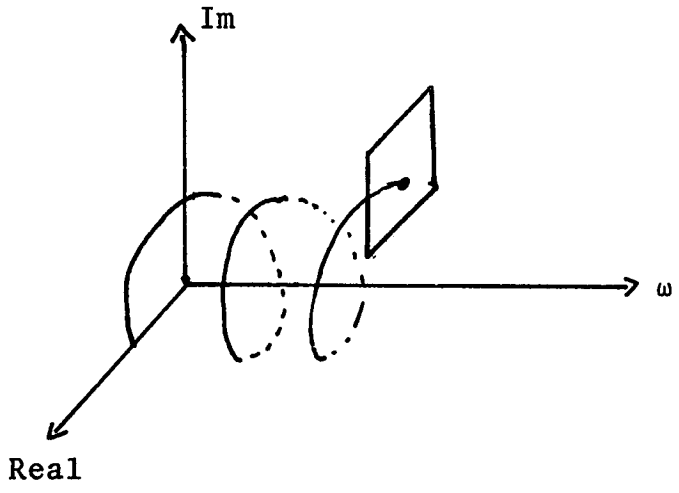


Figure 1-1

If one is doing the general complex Chebyshev approximation problem then one can accept a small phase error and a small magnitude error at a given point, and thereby can avail himself of the possibility of fitting the approximation to the desired function inside a cylinder of radius ε or inside a circle of radius ε at a given frequency (Figure 1-2).

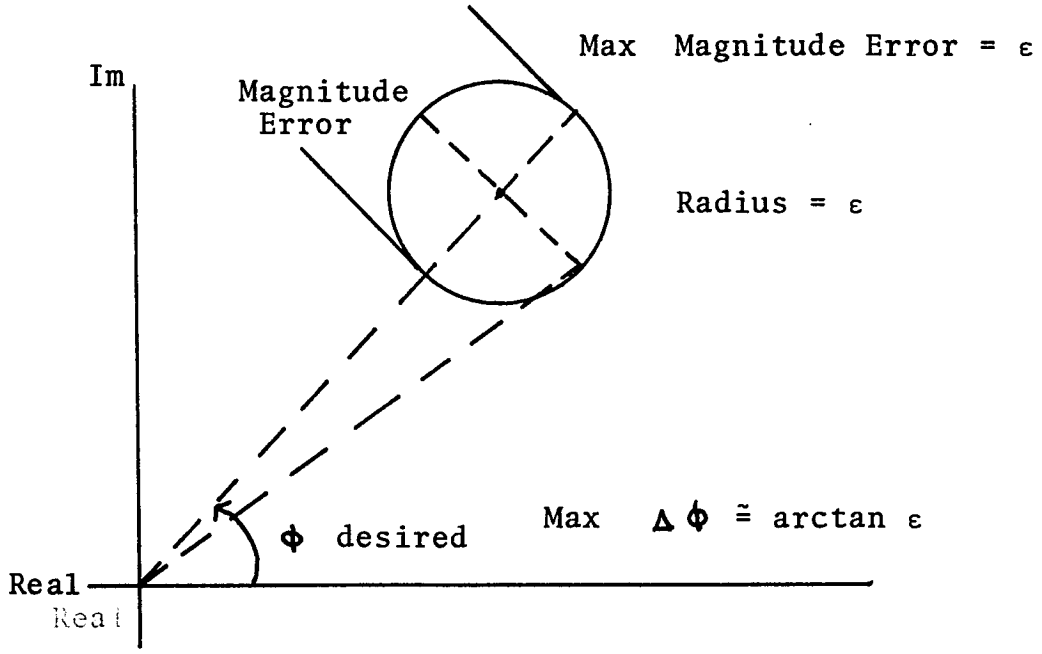


Figure 1-2

It is seen that by trading off a little phase error for a little magnitude error, and vice versa, one can obtain any point lying within the circle of radius ε.

If, however, one fixes the phase to be <u>exactly</u> linear, then one is no longer approximating in a cylinder, but rather on a ribbon lying around the desired function

as shown in Figure 1-3.

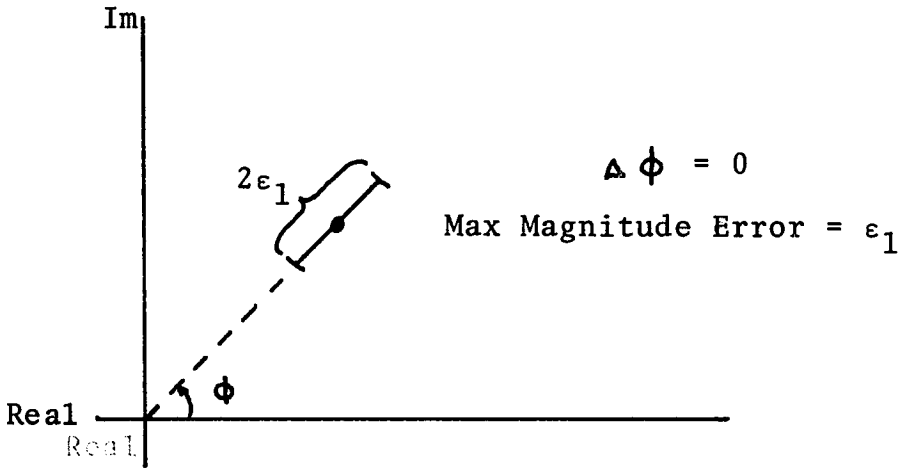


Figure 1-3

Therefore in effect one has constrained the solution to the problem to be in a subspace of the complex approximation problem. Hence, if one is willing to accept a small amount of phase error (or in the case of the optimal magnitude filter any phase at all) then one should be able to obtain a smaller Chebyshev error by taking advantage of the maximum number of degrees of freedom possible. Therefore, this thesis will consider the problem of dropping the linear phase constraint and determining the advantages and disadvantages of doing so.

CHAPTER II

THE OPTIMAL MAGNITUDE FILTER

The optimal magnitude filter is the filter which

$$\underset{\text{all } h_i \,\varepsilon\, \text{Real}}{\text{minimizes}} \quad \{\underset{F \,\varepsilon\, X}{\text{maximum}} \,||D(F)| - |O(F)||\}$$

where

$D(F)$ = desired magnitude response

$$O(F) = \sum_{k=0}^{N} h_k \, e^{-j2\pi kF} = \text{the obtained filter's response}$$

F a normalized frequency variable ε X = [0,1) and h_i i=0,...,N are the filter's coefficients.

Remark 2-1. Although O(F) has a phase associated with its response, the error criteria places no weight on the phase, hence this filter is also termed the optimal magnitude Chebyshev approximation without regard to phase.

Remark 2-2. One notes the following properties of a

transfer function

$$(H(F) = \sum_{k=0}^{N} h_i \, e^{-j2\pi kF})$$

restricted to possessing linear phase.

(a) linear phase $\Leftrightarrow$ symmetric filter coefficients, i.e. $h_{N-i} = h_i$; $i = 0,\ldots,[N/2]$ where $[X]$ denotes the largest integer not exceeding X;

(b) real coefficients imply that the zeros of the transfer function occur in complex conjugate pairs;

(c) symmetric coefficients imply that the transfer function $H(F)$ may be written as (for $N+1 \in$ odd integers)

$$H(F) = e^{-j2\pi nF} \sum_{\mu=0}^{n} d_\mu \cos 2\pi\mu F$$

where $2n+1 = N+1$, $d_o = h_n$, $d_\mu = 2h_{n-\mu}$, $\mu = 1,\ldots, n$, i.e. a strictly imaginary part times a strictly real part;

(d) since

$$\sum_{\mu=0}^{n} d_\mu \cos 2\pi\mu F = d_o + \sum_{\mu=1}^{n} d_\mu \cos 2\pi\mu F$$

the response $H(F)$ may be linearly slid up and down the response axis by changing d_o.

(e) for $c \in$ reals and > 0;

$$c\left[\sum_{\mu=0}^{n} d_\mu \cos 2\pi F_\mu\right] = \sum_{\mu=0}^{n} cd_\mu \cos 2\pi F_\mu$$

which implies that a scaling of all coefficients by c implies a linear scaling of the frequency response.

Proposition 2-1. The procedure for producing the optimal magnitude filter is as follows:

(a) obtain an optimal linear phase Chebyshev approximation of length $2n+1$, $n \in$ positive integers (the method of [4] suffices quite well) to a low pass filter;

(b) add δ_2 (the deviation in the stopband of the length $2n + 1$ linear phase filter) to the center coefficient of that same filter;

(c) find the zeros of this new transfer function

$$\left(H(z) \triangleq \sum_{k=0}^{2n} h_k z^{-k}\right),$$

where the h_k's are the coefficients of the new filter and eliminate all zeros lying outside the unit circle in the z plane, and one each of the double zeros on the unit circle (thereby taking

the square root of the magnitude of the transfer function [see 6, page 93-97] evaluated at $Z = e^{j2\pi F}$);

(d) scale the coefficients of this length n + 1 filter by the factor C = 2/[maximum achieved value of the magnitude of the transfer function

$$H(z) = \sum_{k=0}^{n} d_k z^{-k}\Big|_{Z=e^{j2\pi F}} + \text{minimum achieved}$$

value of the magnitude of the transfer function $H(Z)\Big|_{Z=e^{j2\pi F}}$].

Remark 2-3. In particular if the original length 2n + 1 linear phase filter has a deviation in the passband of δ_1 and in the stopband of δ_2, then one obtains for the length n + 1 optimal magnitude filter pass and stopband deviations of

$$\delta_1' = \left(\frac{2}{\sqrt{1+\delta_1+\delta_2} + \sqrt{1+\delta_2-\delta_1}} \right) \cdot \sqrt{1+\delta_2+\delta_1} - 1.0$$

$$\delta_2' = \left(\frac{2}{\sqrt{1+\delta_2+\delta_1} + \sqrt{1+\delta_2-\delta_1}} \right) \cdot \sqrt{2\delta_2}$$

respectively.

Remark 2-4. It is convenient in the design procedure to be able to express δ_1 and δ_2 of the length 2n + 1 linear phase filter in terms of the resulting δ_1' and δ_2' of the optimal magnitude filter. One may therefore derive the following expressions for δ_1' and δ_2', by essentially reversing the above procedure

$$\delta_1 = \frac{4\delta_1'}{2+2\delta_1'-\delta_2'^2}$$

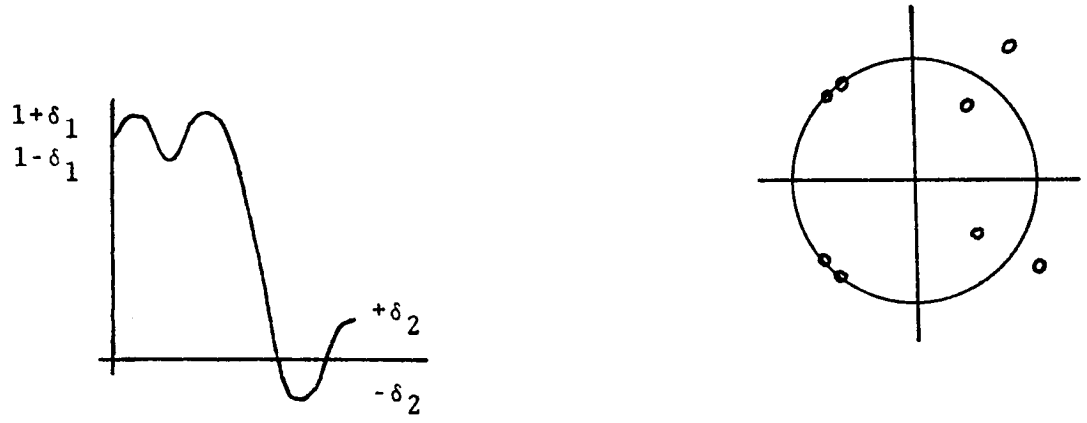
$$\delta_2 = \frac{\delta_2'^2}{2+2\delta_1'^2-\delta_2'^2}$$

Remark 2-5. The procedure of proposition 2-1 is similar to the Hermann Schuessler procedure [7] for producing minimum phase filters, and hence the procedure also produces minimum phase filters. Figure 2-1 shows the zeros locations and resulting alteration in magnitude response as the procedure is applied.

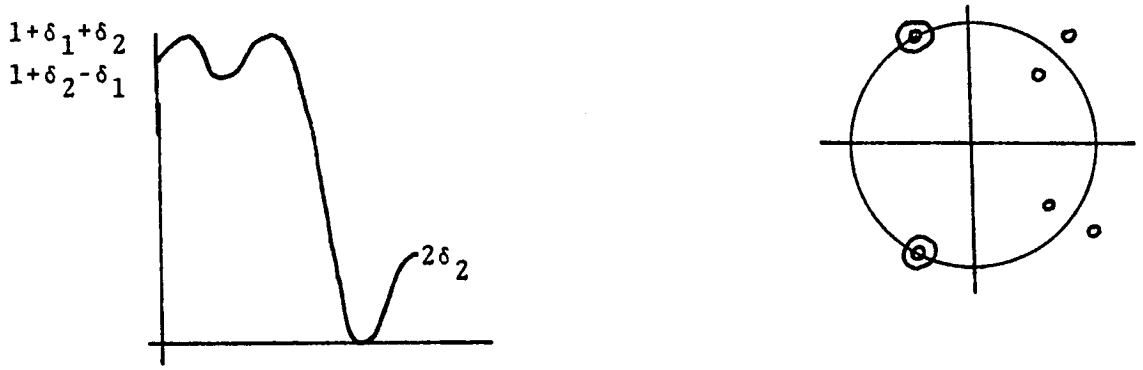
Proof of Optimal Magnitude Property

Lemma 2-1. Given that $\varepsilon_1 < \delta_1$, $0 < \varepsilon_1 < 1$, $0 < \delta_1 < 1$, then

Zeros and Frequency Response of the
Length 2n+1 Linear Phase Filter



Zeros and Frequency Response After Adding δ_2



Zeros and Frequency Response after taking
the Square Root and Scaling

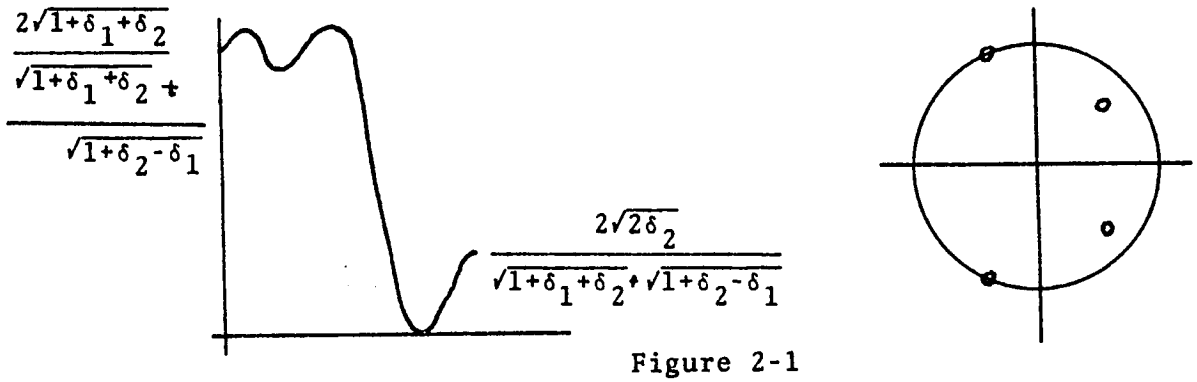


Figure 2-1

(a) $1 + \varepsilon_1^2 + 2\varepsilon_1 < 1 + \delta_1^2 + 2\delta_1$

(b) $1 + \varepsilon_1^2 - 2\varepsilon_1 > 1 + \delta_1^2 - 2\delta_1$

Proof of Lemma 2-1.

(a) $2\delta_1 > 2\varepsilon_1$,

$\delta_1^2 > \varepsilon_1^2 \Rightarrow \delta_1^2 + 2\delta_1 > \varepsilon_1^2 + 2\varepsilon_1$

$1 + \varepsilon_1^2 + 2\varepsilon_1 < 1 + \delta_1^2 + 2\delta_1$ ∎

(b) $\varepsilon_1 + \delta_1 < 2$

$(\varepsilon_1+\delta_1)(\varepsilon_1-\delta_1) > 2(\varepsilon_1-\delta_1)$ multiply by a negative number

$\varepsilon_1^2 - \delta_1^2 > 2(\varepsilon_1-\delta_1)$ rearrange

$1 + \varepsilon_1^2 - 2\varepsilon_1 > 1 + \delta_1^2 - 2\delta_1$ ∎

The proof of the optimal magnitude property follows by contradiction of the fact that an n^{th} order real polynomial of a Chebyshev system can exhibit a maximum of n zeros (see for example [8] page 23).

Proof.

Case I. Assume there exists a filter of length n + 1 with the same cut-off frequencies F_p, F_s as the optimal mangitude filter and possessing a Chebyshev error $\delta_1^{1*} = \delta_1^1$ and a $\delta_2^{1*} < \delta_2^1$ (i.e. a better filter) where the δ_1^1 and δ_2^1 refers to the deviations of the optimal magnitude filter in the pass and stopbands respectively. Reversing the optimal magnitude procedure produces a filter with a δ_1^* and a δ_2^* as a function of δ_1^{1*} and δ_2^{1*} as given in Remark 2-4. In particular, one can reverse the optimal magnitude procedure to produce this linear phase filter with δ_1^* and δ_2^* by:

(a) finding the zeros of the transform function of the length n + 1 filter;

(b) making double zeros of all zeros on the unit circle;

(c) adding a zero outside the unit circle at $(1/d,\theta)$ for each zero inside the unit circle located at (d,θ) (or vice versa) where d is the distance from the origin to the zero, and θ the angle;

(d) multiplying out these zeros and scaling such that one approximates 1 and 0 in the pass and stopband respectively.

In particular one notes that if

$\delta_1'^* < \delta_1'$ and $\delta_2'^* < \delta_2'$ then

$\delta_1^* < \delta_1$ and $\delta_2^* < \delta_2$ (trivially).

Also since one is now back to a length 2n+1 linear phase filter the frequency response H(F) may be written (Remark 2-2), as

$$H(F) = \sum_{\mu=0}^{N} d_\mu \cos 2\pi\mu F.$$

$F \in [0,1/2]$

(F is a normalized frequency variable)

If we associate H(F) with the length 2n + 1 linear phase filter resulting from the optimal magnitude filter and H*(F) with the filter assumed to be better, we may arrive at a contradiction.

If one considers the polynomial obtained by taking the difference of the two error vectors associated with the two filters i.e.

$$P(F) \triangleq e(F) - e^*(F) = D(F) - H(F)$$

$$- (D(F) - H^*(F)) = H^*(F) - H(F)$$

one obtains a Chebyshev polynomial P(F) of order n since H*(F) and H(F) are both Chebyshev polynomials of order n.

Now it is clear that one knows little about the form of the error vector e*(F) associated with the filter assumed to be better except that e*(F) always lies below that of e(F). However, since the optimal magnitude procedure started with the optimal Chebyshev approximation for finite impulse response digital filters with linear phase, one knows that the error vector e(x) exhibits at least n + 2 alternating maxima (see for example [4]). Therefore, P(F) exhibits at least n + 2 sign changes or n + 1 zeros since e*(x) always lies below e(x).

Contradiction: P(F) is a Chebyshev polynomial of order n.

Case II. Assume there exists a filter of length n + 1 with the same cut-off frequencies F_p, F_s as the optimal magnitude filter and possessing a Chebyshev error of $\varepsilon_1' < \delta_1'$, and $\varepsilon_2' < \delta_2'$ where $(\varepsilon_1', \delta_1')$ and $(\varepsilon_2', \delta_2')$ refer to the stop and passbands deviations, respectively. If one considers then doing steps a, b, c as in Case I and multiplying out the zeros, one obtains filters with deviations as shown in Figure 2-2.

If now one (1) adds to the center coefficient of the length 2n + 1 filter resulting from the filter assumed to be better the

$$\min\left[\frac{1}{2} \left(1+\delta_1'^2+2\delta_1'-(1+\varepsilon_1'+2\varepsilon_1'\right), \quad \frac{1}{2}\left(\delta_2'^2-\varepsilon_2'^2\right)\right]$$

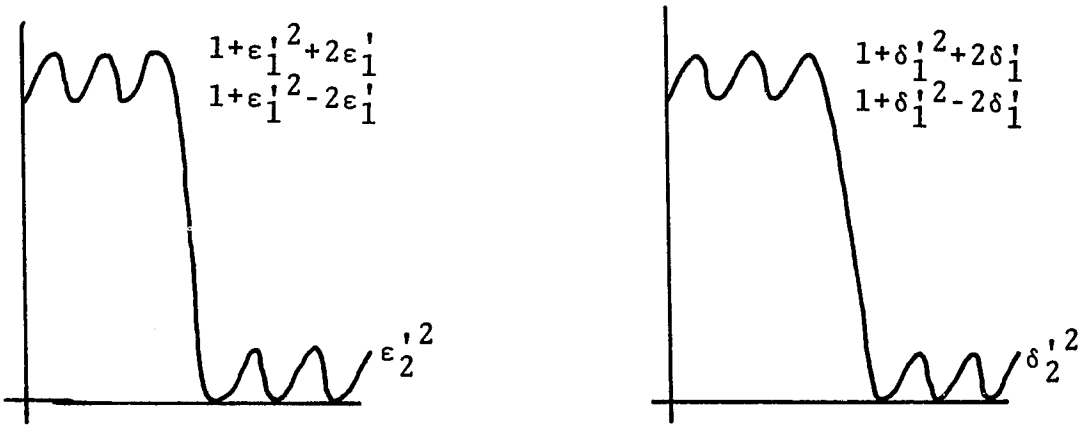


Figure 2-2

(2) shifts and scales both resultant filters by

$$-\frac{1}{2}\delta_2'^2 \quad \text{and} \quad c = \frac{1}{1+\delta_1'^2 - \frac{1}{2}\delta_2'^2} \quad \text{and}$$

(3) again considers the difference of the two polynomials one again obtains a contradiction as in Case I.

Case III. With conditions as before in Case II except assume there exists a filter possessing ε_1^i and ε_2^i such that

$$\varepsilon_1^i < \delta_1^j, \quad \varepsilon_2^j = \delta_2^j$$

where δ_1^j and δ_2^j refer to the optimal magnitude filter's

deviation. Prodeeding as in Case II, we have deviations as shown in Figure 2-3.

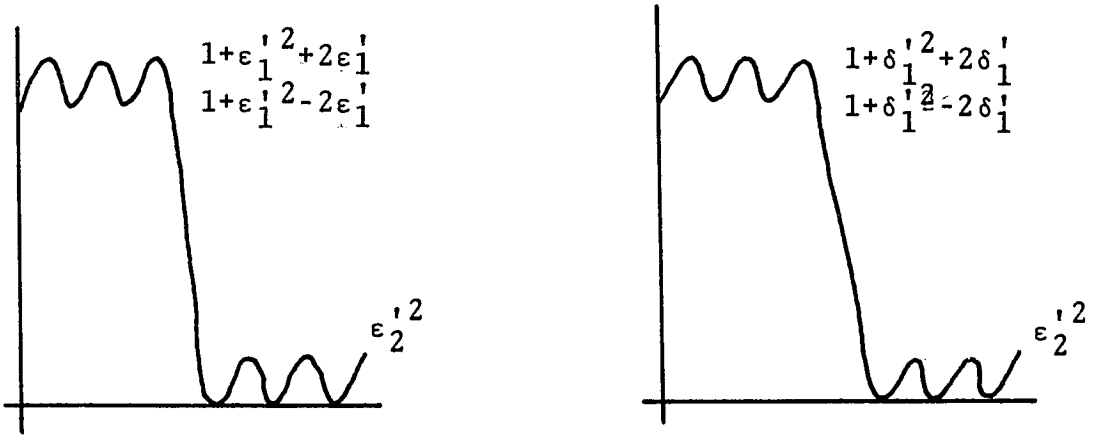


Figure 2-3

From Lemma 2-1

$$1 + \delta_1'^2 + 2\delta_1' > 1 + \varepsilon_1'^2 + 2\varepsilon_1'$$

$$1 + \delta_1'^2 - 2\delta_1' < 1 + \varepsilon_1'^2 - 2\varepsilon_1'$$

If one now (1) scales by k

$$k = \frac{1+\delta_1'^2-2\delta_1'}{1+\varepsilon_1'^2-2\varepsilon_1'} < 1$$

the length 2n + 1 linear phase filter resulting from the filter assumed to be better,

(2) adds to the center coefficient of this same filter

$$\text{the } \min\left[\tfrac{1}{2}(\delta_2'^2 - k\varepsilon_2'^2),\ \tfrac{1}{2}(1+\delta_1'^2+2\delta_1' - k(1+\varepsilon_1'^2 + 2\varepsilon_1')\right]$$

(3) shifts and scales both resultant filters by

$$-\tfrac{1}{2}\,\delta_2'^2 \quad \text{and} \quad c = \frac{1}{1+\delta_1'^2 - \frac{1}{2}\delta_2'^2}$$

and (4) looks at the difference of the two polynomials, one again obtains a contradiction as in Case II.

The Numerical Solution

A computer algorithm was programmed which proceeds in the following manner:

(1) Obtains an optimal Chebyshev linear phase approximation to a low pass filter on a dense grid, through the Parks and McClellan algorithm [4], with the desired passband and stopband cut-offs and the necessary δ_1 and δ_2 as given by Remark 2-4 as a function of the desired δ_1' and δ_2'.

(2) Adds δ_2 (the deviation in the stopband of the length 2n + 1 filter) to the center coefficient of the filter which then implies,

(a) all the zeros in the stopband on the unit circle are double zeros occuring (except for the case $z = e^{j\pi}$) in complex conjugate pairs;

(b) the other zeros must be reciprocal and occur in complex conjugate pairs or lie on the real axis;

(3) Applies a modified Bairstow method of root finding to find the zeros of the transfer function of step 2 and then proceeds to eliminate all the zeros outside the unit circle along with one each of the pair of double zeros located on the unit circle.

Part three deserves further clarification. For a short discussion of the Bairstow's method for finding zeros of a polynomial the interested reader is referred to [10]. The remainder of this discussion will focus on how the starting points for the application of Bairstow's method are found.[1] The algorithm proceeds as follows:

Since one knows the location of the extremal frequencies of the linear phase filter, one can determine with good accuracy the location of the minimas in the stopband and consequentially the location of the double

---

[1]Much of the work with respect to the computer algorithm is due to D. Stahlmach of Rice University, Houston, Texas.

zeros in the stopband, and since the double zeros occur in complex conjugate pairs one has essentially located 4 at a time.

Similarly, the minimas in the passband provide a starting point for the location of the reciprocal pairs of zeros in the passband. In particular the location of the minimas provide a guess as to which angle to assume the zero is located along. The starting guess for the radius is initially taken to be one, with the subsequent starting points for the other zeros the previously found radius of the zero immediately preceeding. The procedure starts with the largest θ and proceeds to smaller θ's. The procedure for pass band zeros is illustrated in Figure 2-4.

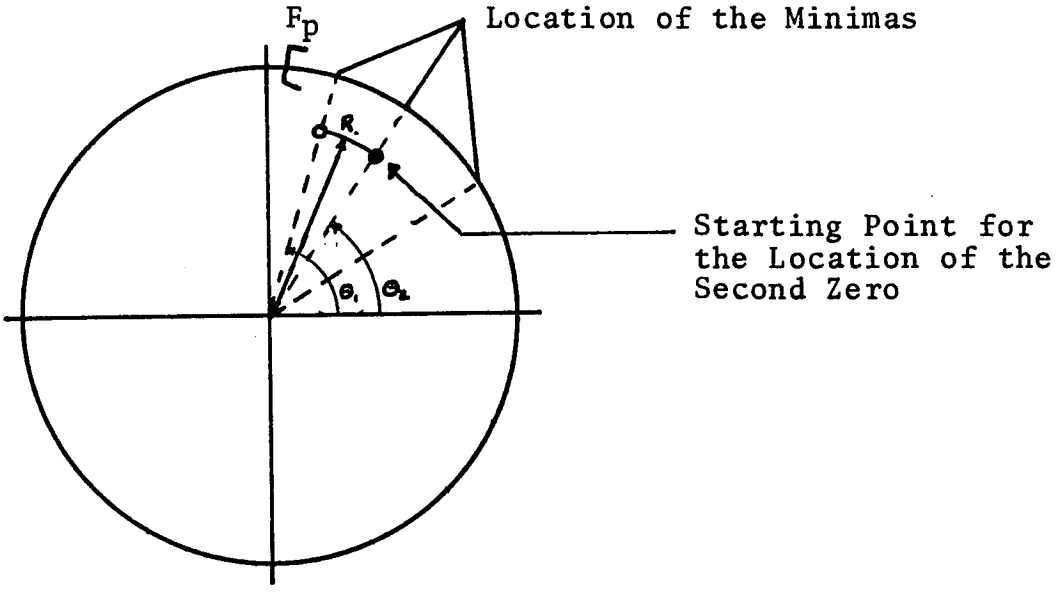


Figure 2-4

Results

This numerical procedure has been applied to many different types of filters with extremely good accuracy and speed. In particular a length 100 optimal magnitude filter has been designed using this procedure.

At this point several meaningful comparisons are apparent.

(1) Given that one fixes N, the length of the filter, how much better can one do using the optimal magnitude filter? i.e. given a linear phase filter of length N and a specified δ_1 and δ_2, how much improvement will an optimal magnitude filter of length N with the same cut-off frequencies, and the same $\delta_1' = \delta_1$ show in δ_2'?

In some cases the improvement is small--i.e., a δ_2' of approximately 6 db below δ_2. However, in other cases, notably in wide band filters, δ_2' can be 20 db below δ_2 for the linear phase filter. An example of the 20 db improvement of the optimal magnitude over the linear phase filter is shown in Figures 2-5 and 2-7.

(2) Perhaps a more significant question is given δ_1, δ_2 and a specified set of cut-off frequencies, what improvement in length can one expect from

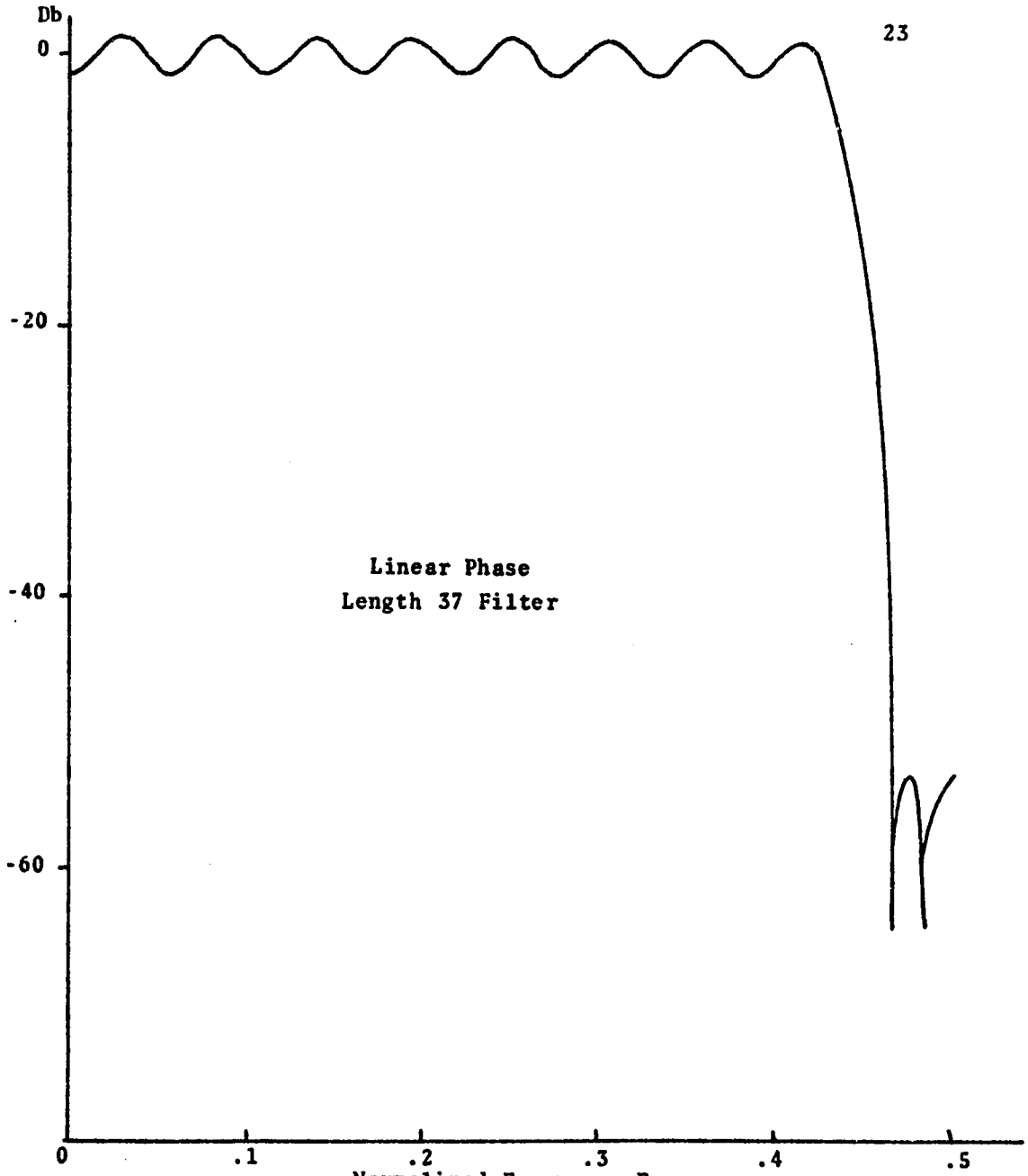

Db
0
-20
-40
-60
23
Linear Phase
Length 37 Filter
0
.1
.2
.3
.4
.5
Normalized Frequency F

Figure 2-5

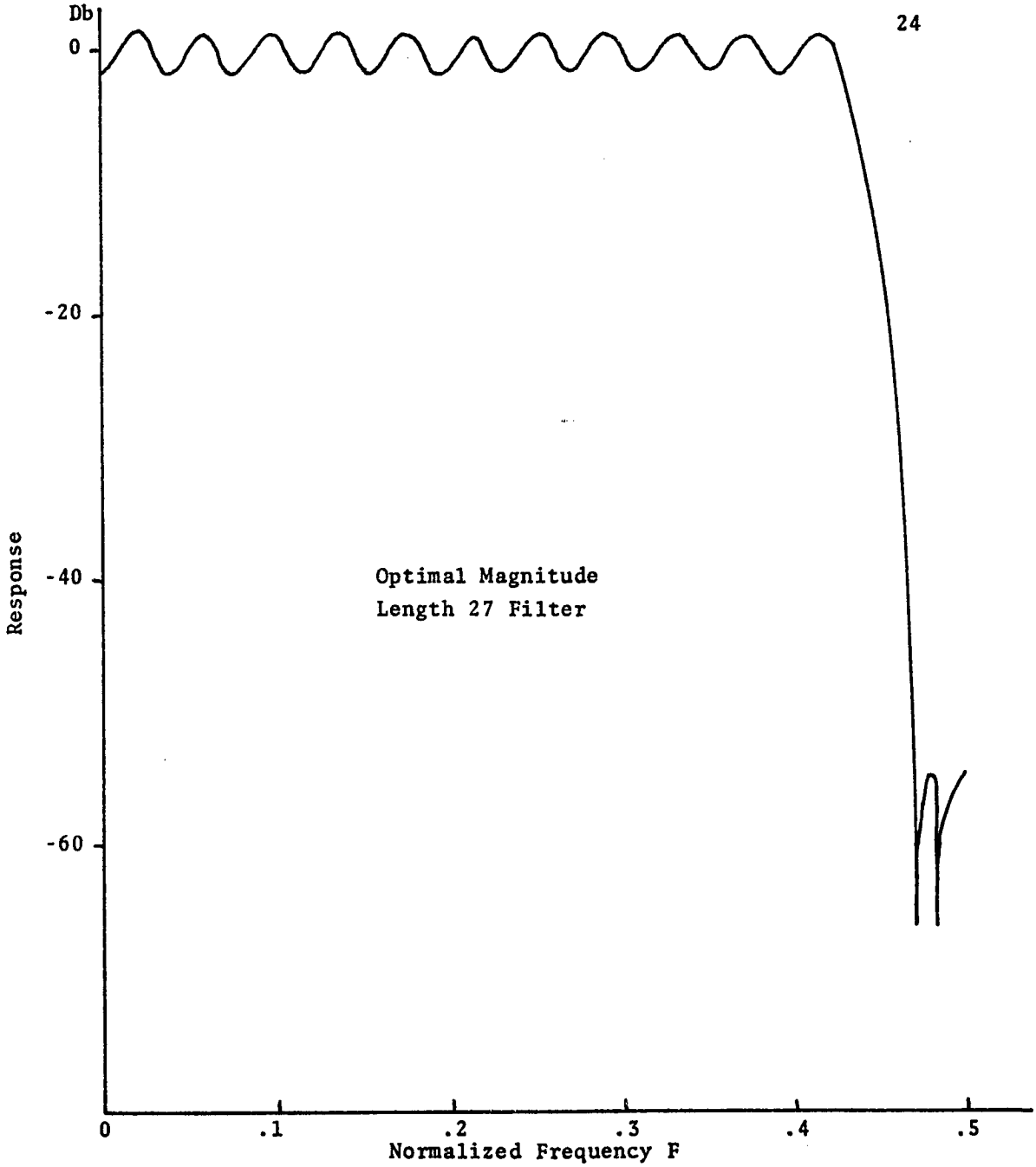

Db
0
-20
-40
-60
Response
24
Optimal Magnitude
Length 27 Filter
0
.1
.2
.3
.4
.5
Normalized Frequency F

Figure 2-6

the optimal magnitude filter over the linear phase filter?

Numerical experience indicates that typically one can expect an improvement in length by a factor of 1/4, i.e. if a length N linear phase filter has a given δ_1 and δ_2, then a length 3/4 N optimal magnitude filter will have the same δ_1, δ_2. To determine in advance for a particular filter what reduction in length one may expect one can compare the values of N found off standard graphs for given deltas (see [11]) to the value of N* determined for the optimal magnitude filter (found by using the same standard graphs and the equations of Remark 2-4).

An example of length reduction is shown in Figure 2-6 where a length 27 optimal magnitude filter exhibits the same magnitude response as the length 37 linear phase filter shown in Figure 2-5.

A table of the impulse responses used to obtain these results is given in Table 2-1.

(3) A third comparison can be made by considering group delay. A linear phase filter of length N always has a delay of (N-1)/2 units, while an

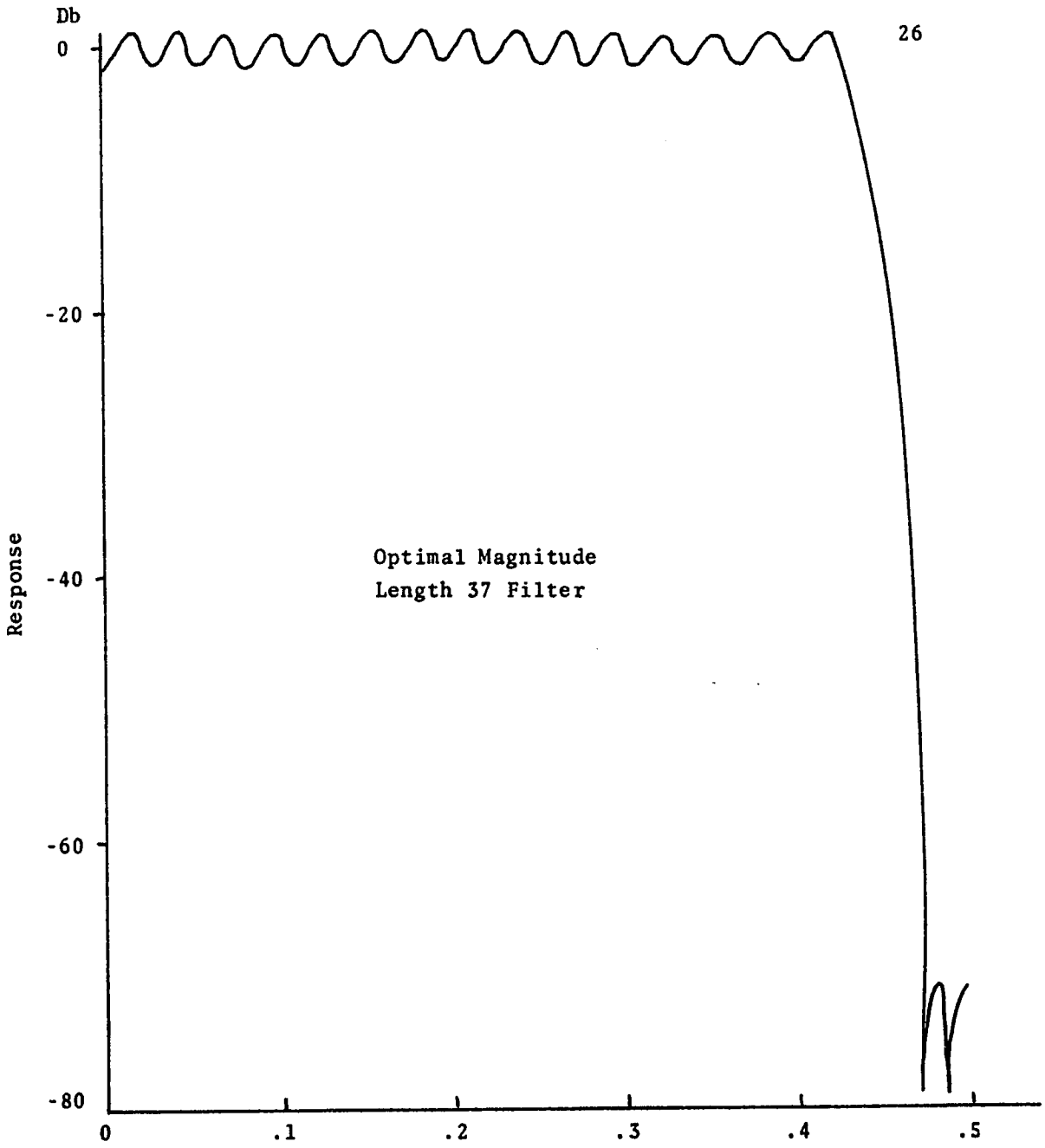

Db
0
-20
-40
-60
-80
Response
26
Optimal Magnitude
Length 37 Filter
0
.1
.2
.3
.4
.5
Normalized Frequency F

Figure 2-7

TABLE 2-1

IMPULSE RESPONSES OF VARIOUS FILTERS USED TO OBTAIN FIGURES 2-5, 2-6, 2-7, 2-8

In All Cases Passband Cutoff=.4375, Stopband Cutoff=.4765625

	Length 37 Linear Phase	Length 37 Optimal Magnitude	Length 27 Optimal Magnitude
1	-.06062266	-.09254351	-.12913201
2	-.02841454	-.11076426	-.07115311
3	-.02473988	.04784192	.05572940
4	-.02244766	-.00922118	-.04363674
5	.02047536	-.01343356	-.03285454
6	-.01792091	.02560267	-.02182491
7	.01406100	-.03071236	.00959521
8	-.00838433	.03083546	.00400265
9	.00064378	-.02710248	-.01835813
10	.00920400	.02020551	.03228686
11	-.02086980	-.01079559	-.04410801
12	.03394659	-.00040782	.05172492
13	-.04771304	.01245042	-.05313543
14	.06140589	-.02400113	.04670824
15	-.07416678	.03357802	-.03146765
16	.08513573	-.03962545	.00758548
17	-.09357668	.04075034	.02329853
18	.09889637	-.03603170	-.05783422
19	.89927420	.02512482	-.09062710
20	.09889637	-.00859818	-.11426923
21	-.09357668	.01197452	.11937226
22	.08513573	.03414077	-.09451642
23	-.07416678	-.05445059	.02685008
24	.06140589	.06906038	.09782017
25	-.04771304	-.07394038	-.29421708
26	.03394659	.06576884	.57732202
27	-.02086980	-.04251300	.62878528
28	.00920400	.00443117	
29	.00064378	.04502167	
30	-.00838433	-.09821310	
31	.01406100	.14243790	
32	-.01792091	-.15913862	
33	.02047536	.12322871	
34	-.02244766	-.00273679	
35	.02473988	-.24135091	
36	-.02841454	.65491806	
37	-.06062266	.54489264	
δ_1	.1520	0.1672	.16908
δ_2	.00237	0.0002892	.00247

optimum magnitude filter, with minimum phase, has considerably less delay.

There is, however, a drawback to the phase characteristics of the optimal magnitude filter. As indicated in Figure 2-8, which is the group delay of the length 27 filter of Figure 2-7, the group delay may prove to be unsatisfactory where little deviation from a constant group delay characteristic is required.

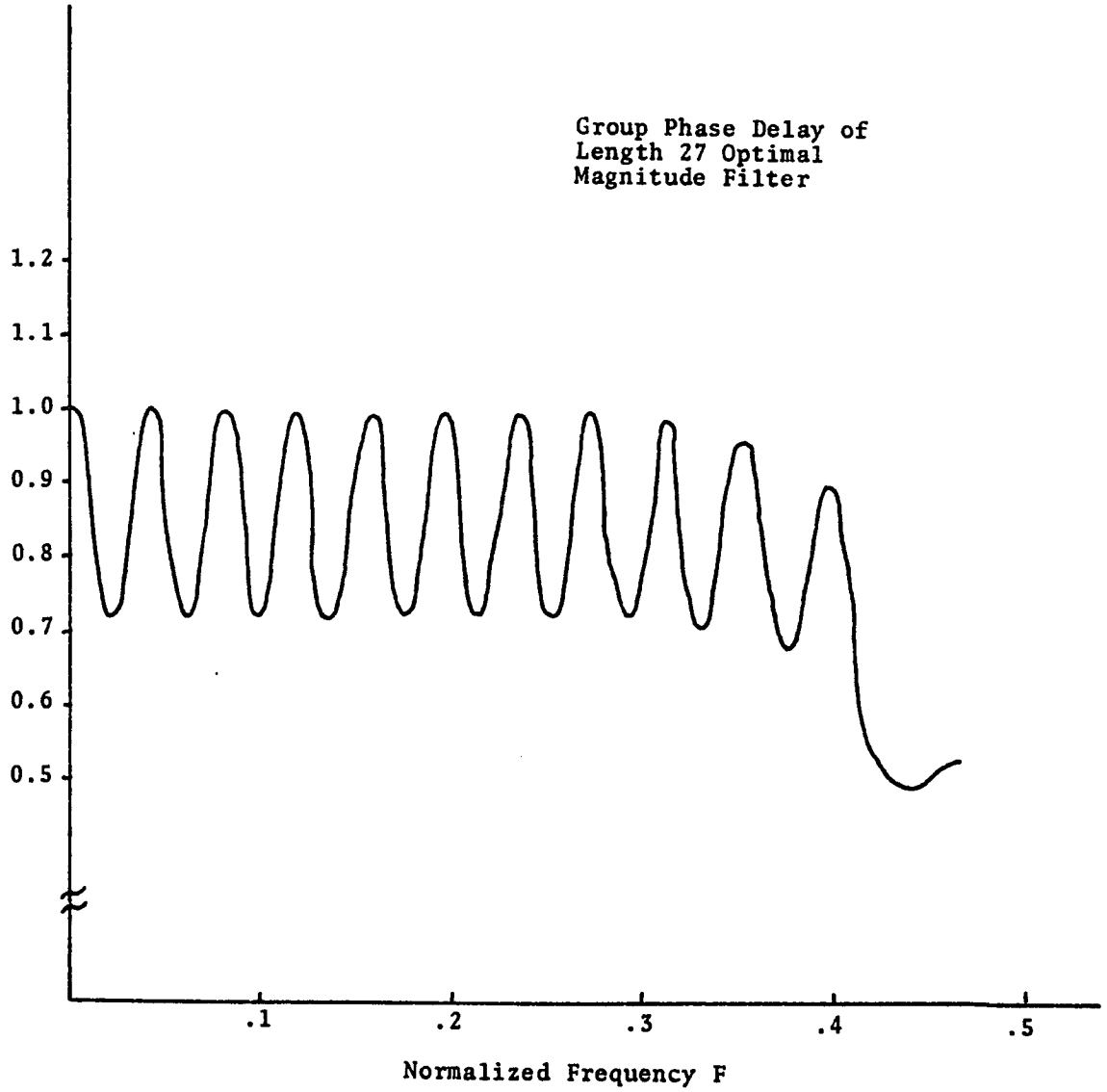

Group Phase Delay of
Length 27 Optimal
Magnitude Filter
1.2
1.1
1.0
0.9
0.8
0.7
0.6
0.5
.1
.2
.3
.4
.5
Normalized Frequency F

Figure 2-8

CHAPTER III

THE OPTIMAL CHEBYSHEV COMPLEX APPROXIMATION

As noted at the end of Chapter II, the optimal magnitude approximation may not be satisfactory if a control of phase slope is desired. Hence one may consider the complex approximation in the hope of

1. obtaining a magnitude response similar to the optimal magnitude, and
2. an approximately linear phase response with a phase slope less than the linear phase since

    a) a better magnitude response implies a better discrimination,
    b) a lower phase slope implies in effect a shorter delay time for the filter (i.e. in real time a pulse placed into the input of such a filter would appear at the output sooner), and
    c) linear phase implies no phase distortion of the signal.

Before proceeding further into the production of the optimal Chebyshev approximation to a desired function (i.e.,

$$\underset{\text{over all } h_i \in \text{Real}}{\text{minimizing}} \quad \{\max_{F \in X} |D(F)-O(F)|\}$$
$$i=0,\ldots,N$$

where D(F) = desired response,

$$O(F) = \sum_{k=0}^{N} h_k e^{-j2\pi kF} = \text{the obtained response}$$

and F a normalized frequency variable) it will be advantageous to present two fundamental results of complex approximation theory. Hence from Lorentz [8].

Proposition 3-1. Let $\Phi = \{\phi_1,\ldots,\phi_N\}$ be a Chebyshev system of complex or real functions on a compact Hausdorff space A that contains at least N+1 points, and let P be a polynomial of best approximation for a continuous function f. Then the set of points $x \in A$ for which $|f(x)-p(x)| = E$ contains at least N+1 points.

Proposition 3-2. For a Chebyshev system there is a unique polynomial of best approximation for each continuous function.

Remark 3-1. In the case of digital filters, Propositions 3-1 and 3-2 imply that a solution exists and is unique where $e^{j2\pi Fk}$, x=0,...,N-1 are the N basis functions, and in terms of previous geometrical arguments the approximating

function must touch the walls of the tube (or ribbon for linear phase approximation) at least N+1 times over the closed interval upon which one is approximating in order to be a candidate for the best approximation.

Remark 3-2. The optimal magnitude filter is optimal in the Chebyshev complex approximation sense also, if the desired function D*(F) is a low pass filter possessing the same passband and stopband cut-offs as the optimal magnitude filter and exactly the same phase.

Proof of Remark 3-2. It is clear that if δ equals the resultant deviation of the optimal magnitude filter, then any candidate for the optimal complex approximation to a low pass filter with that phase must do at least that well, (otherwise it must have done worse and cannot possibly be the best.) Now, assume that the optimal complex Chebyshev approximation to this D*(F) possesses a Chebyshev error of $\varepsilon < \delta$. Then the deviation in the magnitude portion of this filter must be $\leq \varepsilon < \delta$. Contradiction. Therefore the best Chebyshev complex approximation must have a Chebyshev error = δ, and by Proposition 3-2 it must be unique, hence it must be the optimal magnitude filter.

The Lawson Algorithm

The Lawson algorithm will be the method by which the

complex approximations are computed, hence presented below without proof are the basic results of the Lawson algorithm.

It would be desirable if the Lawson algorithm computed the optimal Chebyshev approximation on an interval. Unfortunately, it does not, but instead computes the optimal approximation on a finite point set. This is not a disadvantage, however, if the point set is dense enough. The Lawson algorithm computes the minimum of the

$$\{\text{maximum}\,|D(f)-0(f)|$$

where $f \in \hat{X}$, $\hat{X}$ a finite point set = $[0,1-\varepsilon]$ where $[0,1-\varepsilon]$ merely denotes that the finite point set may contain 0, but not 1, with $D(f)$ and $0(f)$ the desired and obtained responses as before.

The pertinent Lawson results are: (from [9])

NOTATION:

$D(X_i)$ = desired response evaluated at $X_i \in \hat{X}$

$L(A,x)$ = approximating function = $\sum_{i=1}^{n} h_i e^{-j2\pi F(i-1)}$

Let $\omega^k(x)$ $x \in \hat{X}$ be a sequence of weight functions such that

$$\sum_{x\in\hat{X}} \omega^k(x) = 1 \quad \text{and} \quad \{L(A_k,x)\}$$

a corresponding sequence of best weighted least squares approximation to D(x) with weights $\omega^k(x)$. Then

Proposition 3-3. The Lawson algorithm is defined by the recursive relation

$$\omega^{k+1}(x) = \frac{\omega^k(x)\,|D(x)-L(A_k,x)|}{\sum_{x\in\hat{X}} \omega^k(x)\,|D(x)-L(A_k,x)|}$$

where $\omega^1(x)$ is a positive weight function and $L(A_k,x)$ is the best least squares approximation to D(x) with weights $\omega^k(x)$.

Proposition 3-4. Denoting $e^k(x) = D(x) - L(A_k,x)$

$$x \in \hat{X} \text{ and } \sigma^k = \left[\sum_{x\in\hat{X}} \omega^k(x)[e^k(x)]^2\right]^{1/2}$$

then if for some k one has $|e^k(x)|$ constant on $\omega^k(x)$ then $L(A_{k+j},x) = L(A_k,x)$ for all j. Otherwise

$$\sigma^k < \sigma^{k+1} \leq \delta^* = \max_{x\in\hat{X}} |D(x)-L(A^*,x)|$$

where $L(A^*,x)$ is the best Chebyshev approximation to D(x) on $\hat{X}$.

Proposition 3-5. The sequence $\{L(A_k,x)\}$ converges to $L(A_o,x)$ which is a best Chebyshev approximation to $D(x)$ provided $\omega^k(x) \geq \varepsilon > 0$ with $|D(x)-L(A_o,x)| = \sigma^*$ for all $x \ni \omega^k(x) \geq \varepsilon > 0$, and $\sigma^* = \lim_{k\to\infty} \sigma^k$

Remark 3-3. The requirement that $\omega^k(x)$ must remain greater than zero poses no problem numerically since machine round off, etc. will undoubtedly prevent the weight function from ever actually being set equal to zero.

Remark 3-4. Propositions 3-4 and 3-5 are vital since they allow one to determine when to stop the algorithm (i.e. upon the repetition of $|e^k(x)|$.).

Constraints in the Use of the Optimality Criteria

It is clear, that when $e^k(x)$ repeats, one is done, for then he has obtained the optimal Chebyshev complex approximation. It is however, unclear what to do when the error vector $e^k(x)$ does not repeat, as is usually the case when one is designing either moderately long filters or filters on a dense grid.

This, along with the fact that weighted least squares approximation is time consuming and Lawson's algorithm converges very slowly, poses a serious problem. The

problem of $e^k(x)$ not repeating may be dealt with in a number of ways; possibilities include

1. obtaining more accuracy from the computer,
2. allowing the program to run longer, or
3. accepting a small probability that one does not have the optimal filter.

Number 3 is the easiest to obtain. Perhaps the most straightforward way to attain 3 is the following:

a) Run the desired filter on a very coarse grid. (This will usually imply a quick convergence where the error repeats itself, with some Chebyshev error δ.)

b) Run the desired filter again on a dense grid for a suitable length of time (i.e. to where the error is changing sufficiently slowly to indicate further iterations are returning little), and again obtain a Chebyshev error $\varepsilon \geq \delta$.

If $\varepsilon = \delta$, all is well, since one then knows that he has obtained the optimal Chebyshev approximation. However, in most cases $\varepsilon > \delta$ and then one can only say (if the error vector $e^k(x)$ contains at least N+1 maxima where N is the length) that one has a candidate for the optimal complex approximation that can not be any more than $\rho = |\varepsilon - \delta|$ away from the optimal. Hopefully, $|\varepsilon - \delta|$ will

be small.

The Numerical Solution and Results

The Lawson algorithm computes the optimal complex approximation as a limit of successive best least squares approximations modified by an appropriate weighting function. Therefore, it is necessary that one place the complex approximation problem in least squares terms. In effect one has an over determined set of equations

$$Ah = D$$

where h is a column vector (n,1) representing the impulse response of the filter,

D is the desired frequence response at M points (m,1), and

A is the coefficient matrix whose (i,k) element is

$$e^{-j(k-1)((i-1) \times \frac{2\pi}{M})}$$

$$i = 1,\ldots,M, \quad k=1,\ldots,N.$$

With this notation the best weighted least squares approximation is given by

$$h^* = (A^H Q A)^{-1} A^H Q D$$

where A^H implies A transposed and conjugated and Q is a m x m weighting matrix with diagonal entries only. The above solution is the one programmed on the computer.

One notes that in order for h* to be real, D must have even magnitude and odd phase and must be over the interval [0,1) (Q real). Hence in terms of the optimality criteria, one must have at least N + 1 maxima of the error vector on [0,1).

In order to investigate the Lawson algorithm, the following two cases are considered. In one case the algorithm is required to design a linear phase filter of length 9 with cut off frequencies of F_p = cut off frequency in the passband of 0.2 and F_s = cutooff frequency in the stopband of 0.3. The resultant coefficients are then compared to coefficients derived from a standard design method for linear phase filter [12]. As seen in Table 3-2, the coefficients agree quite well, and there is little question that Lawson is capable of designing linear phase filters. In the second case a length 5 optimal magnitude filter is obtained using the method of Chapter II with approximately unit weighting. The F_p and F_s are .15 and .25 respectively for this filter. Remark 3-2 is then applied and the Lawson algorithm is required to approximate $1,\theta$ in the passband (where 1 denotes the magnitude and θ the optimal magnitude's phase response) and 0.0 in the stopband. If the Lawson algorithm is working properly it should converge to the optimal magnitude filter's

coefficients. Table 3-2 demonstrates that again the Lawson algorithm has performed satisfactorily. The small differences are easily attributable to

1. the optimal magnitude filter did not have exactly unit weighting or
2. differences in grid size.

With the Lawson algorithm performing well at both ends of the scale (i.e., linear phase, no phase) one should now consider the case of the middle ground-time complex approximation.

In order to demonstrate that the complex approximation filter performs as indicated at the start of the chapter, a filter of length 11 with F_p = 0094 F_s = .152 with a slope factor of 0.7 (where 0.7 denotes 0.7 of 1 where 1.0 is defined as equaling conventional linear phase i.e, a phase of $\theta = 2\pi F \cdot ((N-1)/2)$ radians where N=length, and F is the normalized frequency variable).

The result is shown in Figure 3-1 along with an optimal magnitude filter. In particular the resulting δ's are: for the complex approximation filter $\delta_1 = \delta_2 \tilde{=} .147$, for the optimal magnitude filter $\delta_1 \tilde{=} \delta_2 \tilde{=} .140$. If the complex approximation is to be useful it must demonstrate a control over the phase. This is indeed the case as indicated in Figure 3-2 where the complex group delay varies considerably less than the

TABLE 3-1

COEFFICIENTS OF LINEAR PHASE FILTERS

	Standard Method	Lawson's Method
1	0.00000	0.00000
2	-0.11960	-0.11912
3	0.00000	0.00000
4	0.31312	0.31409
5	0.49999	0.50000
6	0.31312	0.31409
7	0.00000	0.00000
8	-0.11960	-0.11912
9	0.00000	0.00000

TABLE 3-2

COEFFICIENTS OF OPTIMAL MAGNITUDE FILTERS

	Standard Method	Lawson's Method
1	-0.04096	-0.03041
2	0.22048	0.22622
3	0.36749	0.36966
4	0.38880	0.38566
5	0.27827	0.27264

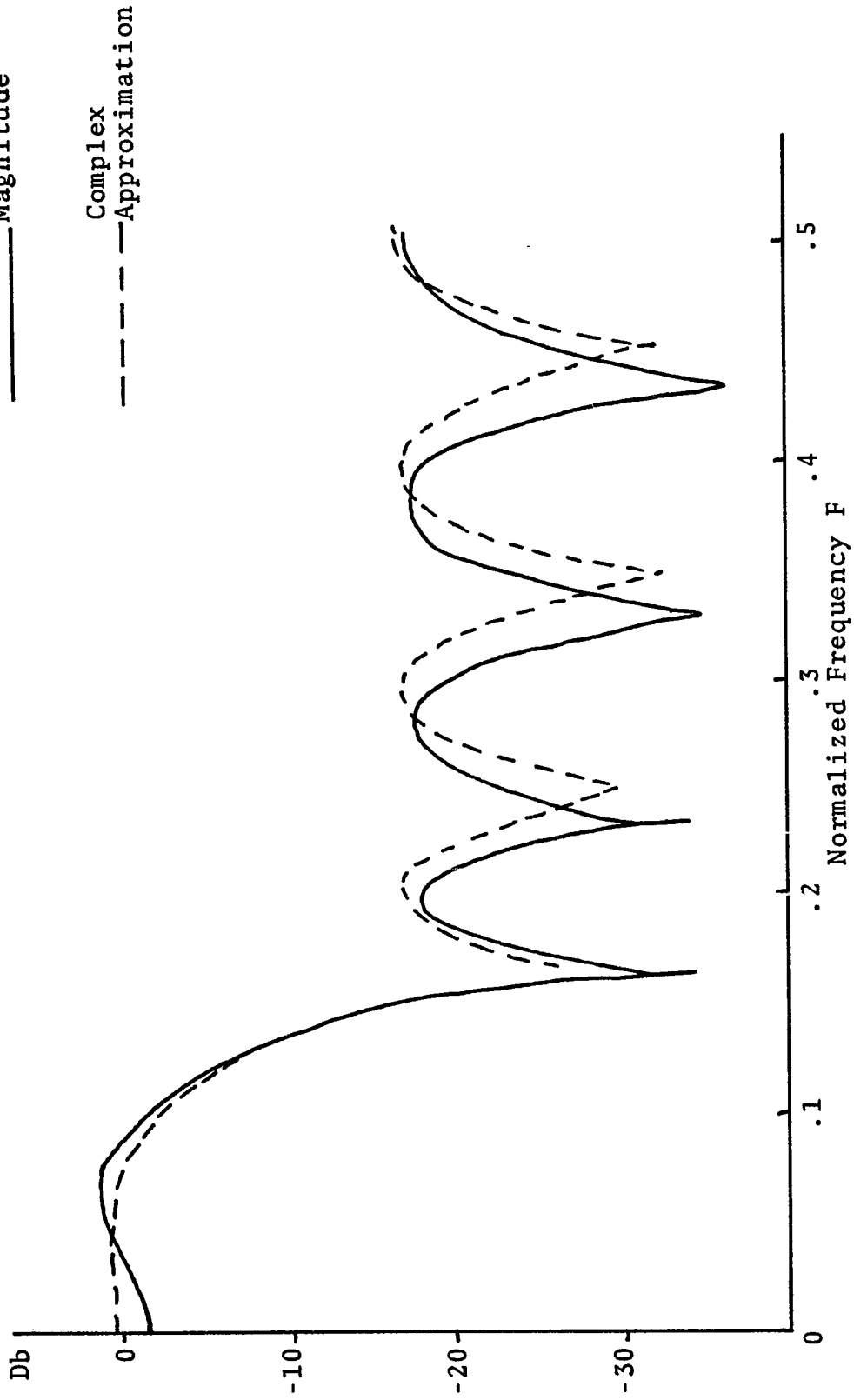

Optimum Magnitude
Complex Approximation
Db
0
-10
-20
-30
.1
.2
.3
.4
.5
Normalized Frequency F

Figure 3-1

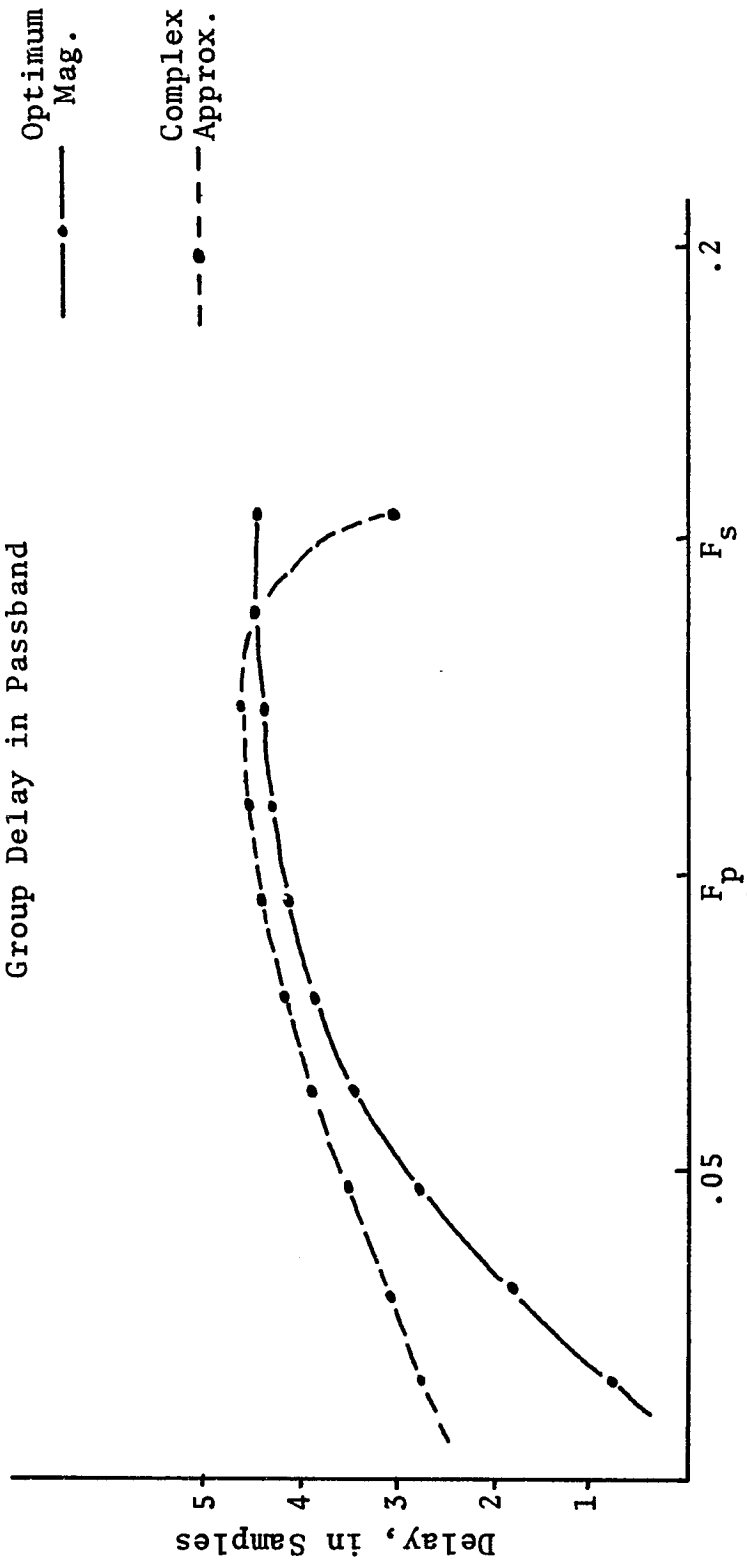

Optimum Mag.
Complex Approx.
Group Delay in Passband
Delay, in Samples
1
2
3
4
5
.05
F_p
F_s
.2

Figure 3-2

optimal magnitude resultant group delay.

Remark 3-5. The linear phase filter possessing the same attributes as the other two filters of Figure 3-1 has a $\delta_1 = .14$ and a $\delta_2 = .19$.

Remark 3-6. Another possible use of the complex approximation filter is in the area of phase compensation. The Lawson algorithm will accomplish this now, but the usefulness is limited due to the short length requirements of Lawson and the slow convergence.

CHAPTER IV

COMPARISONS OF LINEAR PHASE, OPTIMAL MAGNITUDE AND COMPLEX APPROXIMATION FILTERS

Several comparisons between the various filters have already been made (see for example pages 22-28 and pages 38-43), however a comparison implied, but yet to be made is in the differing output delay times of the three filters to a pulse input. Heuristically, one would expect that in response to a pulse input, the optimal magnitude (also minimum phase) would respond the quickest (although a hard statement of this quality is lacking since minimum phase does not necessarily imply a minimum phase slope), followed by the complex approximation filter with a phase slope approximation to a linear slope with a slope factor less than normal linear phase, and finally the linear phase filter which always has a delay of (N-1)/2 sample time intervals where N is the length of the filter. This comparison is demonstrated in Figure 4-1 where a sampled Gaussian pulse is placed into each of the 3 different filters (all of length 9 with the same cut off frequencies and weighting factor). The results are exactly as predicted; i.e., the pulse peaks first in the output of the optimal magnitude,

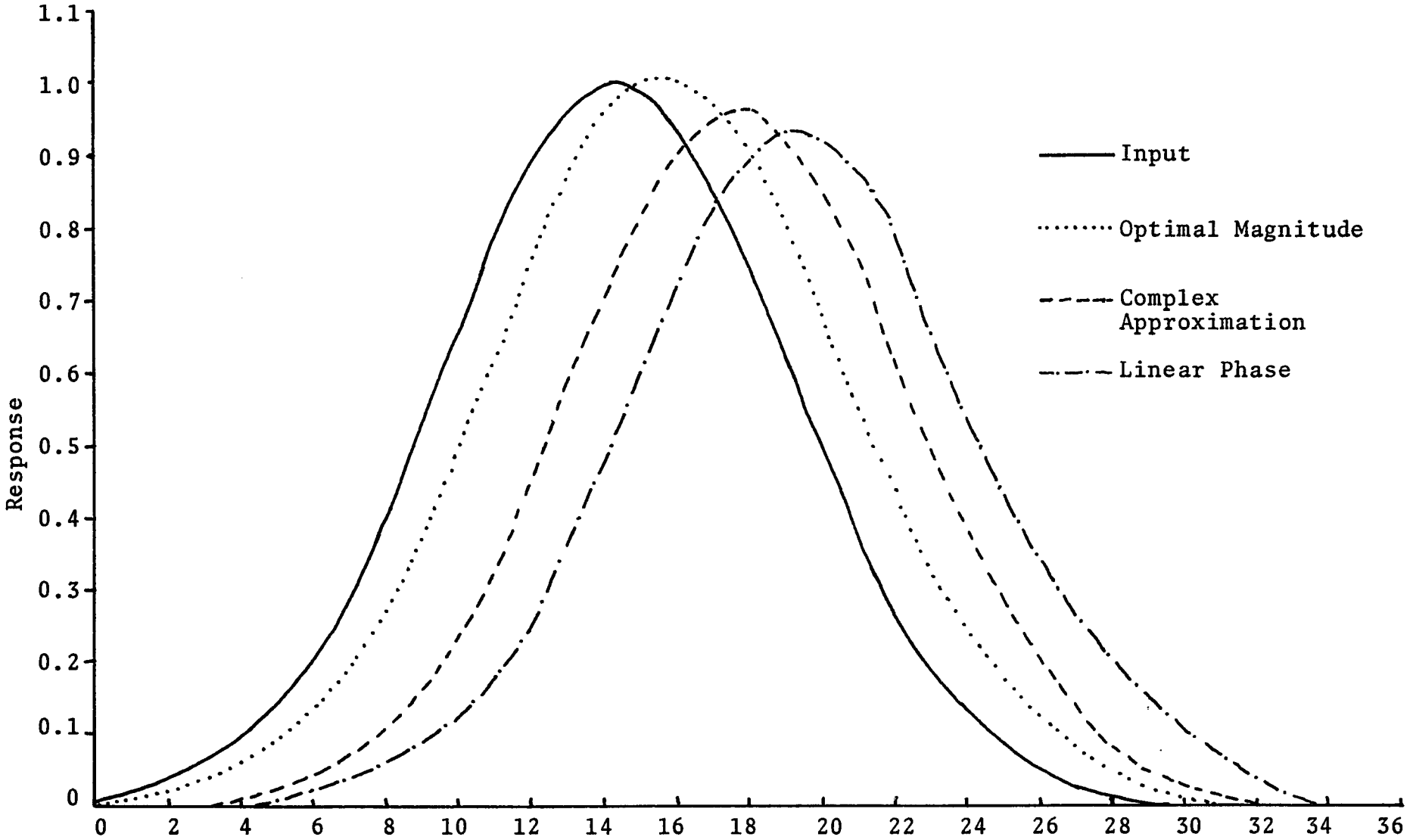

1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Response
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36
Time (# of sample intervals)
Input
Optimal Magnitude
Complex Approximation
Linear Phase

Figure 4-1

followed by the complex approximation filter and then the linear phase filter.

In comparing the three filters, optimal magnitude (O.M.), complex approximation (C.A.) and linear phase (L.P.) one is drawn to the interesting manner in which the optimal magnitude and linear phase represent two optimal extremes with the optimal complex approximation representing the optimal answer in the middle ground. In particular one notes the outstanding attribute of linear phase filters--no phase distortion. One also notes the outstanding attribute of the optimal magnitude filter, i.e. it obtains the lowest Chebyshev error of any finite impulse response digital filter possessing its cut off frequencies and weighting. It is interesting to note that the complex approximation filter possesses attributes of both and the outstanding properties of neither (i.e. some control over phase, but not linear, some improvement of the Chebyshev error over linear phase, but not the best), hence, it is a true compromise between the two extremes.

Therefore, in the simplest terms one should consider

a) using the L.P. filter when a rigid constant group delay characteristic is required,
b) using the O.M. filter where no phase constraints are required and
c) using the C.A. filter for the cases in between.

CHAPTER V

CONCLUSIONS

In summary one concludes that in some cases (notably wide band filters) one can obtain a large improvement in magnitude response or in the length filter required to meet a given set of magnitude specifications, if one is willing to accept no control over the variation of the group phase delay through the use of the optimal magnitude filter. In particular for length improvement one can apply the rule of thumb that for a given δ_1, δ_2 (deviations in the pass and stopbands) the optimal magnitude filter has a length of approximately three-fourths of the linear phase filter producing these same δ_1, δ_2. For δ_1', δ_2' (the resultant pass and stopband deviations for the optimal magnitude filters) improvement one can determine the δ_1' and δ_2' for the length n + 1 optimal magnitude filter given the δ_1 and δ_2 of a length 2n + 1 linear phase filter by applying the formulas

$$\delta_1' = \left(\frac{2}{\sqrt{1+\delta_1+\delta_2}-\sqrt{1+\delta_2-\delta_1}} \right) \cdot \sqrt{1+\delta_1+\delta_2} - 1$$

$$\delta_2' = \left(\frac{2}{\sqrt{1+\delta_2+\delta_1}+\sqrt{1+\delta_2-\delta_1}} \right) \cdot \sqrt{2\delta_2}$$

One also notes that for any given filter possessing a specified pass and stopband cutoff and a specified weighting factor, the Chebyshev deviation obtained by the optimal magnitude filter's deviation constitutes the smallest possible Chebyshev error obtainable.

In summary the optimal Chebyshev complex approximation filter performs as predicted, i.e. it possesses

a) a Chebyshev error less than that obtained by the optimal magnitude filter and greater than that obtained by the linear phase filter for normal specifications of desired linear phase with a slope less than the slope obtained by the linear phase filter;

   1. hence the optimal complex approximation presents a magnitude deviation less than the linear phase, and
   2. in general a phase distortion curve better than the optimal magnitude filter;

b) a lower phase slope which implies in effect a shorter delay time for the filter (i.e. in real time a pulse placed into the input of such a filter would appear at the output sooner).

One also notes that while Lawson's algorithm is exceptionally slow and generally unsuited for filter design

due to length considerations, it provides the only known means of obtaining an optimal complex Chebyshev approximation.

Hence future work must be directed toward the production of a fast design algorithm for optimal Chebyshev approximation.

CHAPTER VI

APPENDIX A

OPTIMAL MAGNITUDE FILTER COMPUTER PROGRAM

```
      IMPLICIT REAL*8 (A-H,O-Z)
      DEFINE FILE 8(1,527,U,IXYZ)
C     PROGRAM TO DETERMINE THE ZERO LOCATIONS OF THE FINITE IMPULSE
C     RESPONSE OPTIMAL MAGNITUDE DIGITAL FILTER USING THE
C     BAIRSTOW METHOD OF FINDING ZEROS AND THE J.D. FISHER METHOD FOR
C     THE PRODUCTION OF THE OPTIMAL MAGNITUDE FILTER
C
C     J.D. FISHER, T.W. PARKS, D. STAHLMACH OF RICE UNIVERSITY
C     HOUSTON,TEXAS -- JANUARY 1, 1973
C
C     FROM THE DESIGN OF AN OPTIMAL LINEAR PHASE FILTER DESIGN PROGRAM
C     THE PROGRAM REQUIRES AS AN INPUT THE FOLLOWING PARAMETERS
C          A = LINEAR PHASE FILTER'S COEFFICIENTS
C          FOPT = EXTREMAL FREQUENCIES OF THE LINEAR PHASE FILTER(LPF)
C          NF = LENGTH OF LPF
C          KR = NUMBER OF PASSBAND RIPPLES OF LPF
C          KRP = NUMBER OF STOPBAND RIPPLES OF THE LPF
C          TW = TRANSITION WIDTH OF THE LPF
C          AA = WEIGHTING FACTOR (='S PASSBAND DEVIATION/STOPBAND
C                    DEVIATION)
C          RHO = DEVIATION IN STOPBAND
C          R1 = DEVIATION IN PASSBAND
C          FP = PASSBAND CUTOFF
C          FS = STOPBAND CUTOFF
C               WHERE FP,FS,FOPT ETC. ARE GIVEN IN TERMS OF A
C               NORMALIZED FREQUENCY VARIABLE = (0.,.5)
C          THIS INFORMATION IS SET UP TO BE READ OFF A DISK FILE  IN A
C          SEQUENTIAL MANNER.  HENCE THE MATRICES A AND FOPT
C          MUST HAVE THE SAME DIMENSIONS IN BOTH PROGRAMS IF THIS
C           INPUT SCHEME IS TO BE USED
C
C     THIS PROGRAM ALSO REQUIRES THE INDEPENDENT INPUT BY CARDS
C      OF
C          CARD ONE - = NL = THE LENGTH OF THE LPF IN THE FIRST THREE
C          COLUMNS
C          CARD TWO = LL = NUMBER OF ITERATIONS IN FIRST THREE COLUMNS
C          THIS NUMBER IS GENERALLY SET TO ABOUT THIRTY
C          CARD 3 = NPOINT = NUMBER OF POINTS AN FFT OF THE
C          RESULTING OPTIMAL MAGNITUDE FILTER IS TO BE CALLED ON.
C          IT MUST BE A POWER OF 2
C     HENCE A SAMPLE CARD INPUT WOULD BE
C
C     021
C     030
C     128
C
C     SIGNIFYING A LENGTH 21 FILTER WITH A MAXIMUM
C     NUMBER OF ITERATIONS OF 30 WITH A FFT CALLED ON 128 POINTS
C
C     THE OUTPUT CONSISTS OF
C
C     (1) THE COEFFICIENTS OF THE LPF AND THE COEFFICIENTS
```

```
C      OF THE LPF AS THEY WOULD BE ASSUMING THE ZERO LOCATIONS AS
C      FOUND BY THE PROGRAM --- IN BETWEEN THESE PRINTOUTS THE
C      VARIOUS REMAINDER VALUES ARE PRINTED OUT INDICATING HOW
C      WELL THE ZEROS HAVE BEEN FOUND
C
C      (2) THE OPTIMAL MAGNITUDE FILTER'S COEFFICIENTS
C
C      AND(3) THE FFT OF THE OPTIMAL MAGNITUDE FILTER
C
C
C      THE PROGRAM IS CURRENTLY SET UP TO HANDLE A MAXIMUM
C      LENGETH FILTER OF 128, IF AN INCREASE IN LENGTH IS DESIRED THE
C      MATRIX SIZE OF THE MATRICIES CURRENTLY HAVING DIMENSION 128 MUST
C      BE CHANGED TO EQUAL OR EXCEED THE NEW LENTH DESIRED
C      THE OTHER MATRICES SHOULD BE AT LEAST 1/3 THE SIZE
C      OF THE NEW DIMENSION, WITH THE EXCEPTION
C      OF THE MATRICIES USED IN THE FFT CALL   - WHICH ARE INDEPENDENT
C      SIMILARLY THE DATA INITIALIZATION STATEMENTS MUST BE CHANGED
C
       COMMON/LI/A(128),F,N
       COMMON/QB/ASUB(128),FSUB,NSUB
       COMMON/FI/RR,KSP
       COMMON/BA/EB(100),E(100),EC(100),X(100),Y(100),XC(100),
      $ YC(100),KS
       COMMON/MB/ZETA,ETA,RP,LL
       COMMON/CD/EA(100),FA(100),GA(100),WA(100),ZA(100),WAC(100),
      $ ZAC(100),KP
       COMMON/PR/XX(128),IN
       DIMENSION FOPT(128),SMIN(128),PMIN(128),RAA(128),ZOZ(100)
       DIMENSION RFA(100),AAA(128),BB(128),RBB(100),
      $ SFAC(100),PFAC(100)
       DIMENSION ROPO(128),REPO(128),RTPO(30),ASPL(128)
       DIMENSION PSR(128),POR(128),AAR(128),RGA(100),AAL(128)
       DATA
      4 PMIN/128*0.0/,RAA/128*0.0/,ZOZ/100*0.0/,RFA/100*0.0/,AAA/128*
      5 0.0/,BB/128*0.0/,RBB/100*0.0/,SFAC/100*0.0/,PFAC/100*0.0/
       DATA ROPO/128*0.0/,REPO/128*0.0/,RTPO/30*0.00/,ASPL/128*0.0/,
      1 PSR/128*0.0/,POR/128*0.0/,AAR/128*0.0/,RGA/100*0.0/,AAL/128*
      2 0.0/
       DO 881 I = 1,128
       A(I) = 0.0
       ASUB(I) = 0.0
       XX(I) = 0.0
       FOPT(I) = 0.0
       SMIN(I) = 0.0
  881  CONTINUE
       DO 882 I = 1,100
       EB(I) = 0.0
       E(I) = 0.0
       EC(I) = 0.0
       X(I) = 0.0
       Y(I) = 0.0
       XC(I) = 0.0
```

```
      YC(I) = 0.0
      EA(I) = 0.0
      FA(I) = 0.0
      GA(I) = 0.0
      WA(I) = 0.0
      ZA(I) = 0.0
      WAC(I) = 0.0
      ZAC(I) = 0.0
  882 CONTINUE
      NWI = 0
      NTR = 0
      NWR = 0
      NWT = 0
    1 FORMAT(I3)
    2 FORMAT(F15.8)
    3 FORMAT(/5X,'ENTER ORDER FILTER(I3)')
    4 FORMAT(/5X,'IMPULSE RESPONSE'/(E20.12))
    7 FORMAT(/5X,'DEGREE OF POLYNOMIAL N=',I3)
    8 FORMAT('FILTER LENGTHS DISAGREE')
   16 FORMAT(/5X,'ENTER NUMBER OF ITERATIONS(I3)')
   17 FORMAT(/'=PTS.(I3)')
      KS=0
      PRINT3
      READ1,NL
      PRINT 16
      READ1,LL
      PRINT 17
      READ1,NPOINT
      NN=(NL-1)/2
       READ(8'1) A,FOPT,NF,KR,KRP,TW,AA,RHO,R1,FP,FS
      F = A(1)
      NNP1 = NN+ 1
      DO 888 III = 2,NNP1
  888 A(III-1) = A(III)
      IF(NF-NL)14,12,14
   12 A(NN)=A(NN)+RHO
      N=2*NN
      A(N)=F
      NNM1 = NN - 1
      DO 11 I = 1,NNM1
   11 A(N-I)=A(I)
      PRINT7,N
      PRINT4,F,(A(I),I=1,N)
      ETA = 8.0D0 * DATAN(1.0D0)
      KWICH=3
      NP2 = NN+2
      DO 30 J = 1,NP2
      IF(FOPT(J)-FS)30,31,31
   31 JS=J
      GO TO 32
   30 CONTINUE
   32 IS=0
      IF(JS.EQ.(NN+2)) GO TO 45
```

```
      KP1=JS+1
      NP2=NN+2
      DO 40 K = KP1,NP2,2
      IS=IS+1
   40 SMIN(K-JS+1-IS)=FOPT(K)
      IF(SMIN(IS).LT.FOPT(NN+2)) GO TO 47
      DO 60 K=1,IS
      IF(SMIN(K).GT.(0.5-0.5E-08)) GO TO 63
      ZETA=SMIN(K)
      KS=KS+1
      CALL MILBA
   60 CONTINUE
      GO TO 55
   63 IS=IS-1
      KS =KS+1
      ZETA=0.5
      RP=1.0
      KWICH=1
      NITCH=1
      CALL REIM(EB,E,EC,X,Y,KS)
      GO TO 55
   47 DO 50 K=1,IS
      ZETA=SMIN(K)
      KS=KS+1
      CALL MILBA
   50 CONTINUE
      IF(FOPT(NN+2).GT.(0.5-0.5E-08)) GO TO 55
      KS=KS+1
      ZETA=0.5
      RP=1.0
      KWICH=1
      NITCH=1
      CALL RECIP(EB,E,EC,X,Y,KS)
      GO TO 55
   45 KS=KS+1
      ZETA=0.5
      RP=1.0
      KITCH=1
      NITCH=1
      CALL RECIP(EB,E,EC,X,Y,KS)
   55 NSBR=IS*4+(KS-IS)*2
      KP=0
      IP=0
      IF(FOPT(JS-1).EQ.FOPT(1)) GO TO 85
      IF(FOPT(JS-2).EQ.FOPT(1)) GO TO 67
      DO 65 K=1,100
      KN=JS-1-2*K
      IF(KN.LT.1) GO TO 70
      IP=IP+1
   65 PMIN(K)=FOPT(KN)
   70 RP=1.0
      IF(PMIN(IP).GT.FOPT(1)) GO TO 77
      DO 80 K=1,IP
```

```
      IF(PMIN(K).LT.0.5E-10) GO TO 87
      ZETA=PMIN(K)
      KP=KP+1
      CALL LINBA
   80 CONTINUE
      GO TO 85
   87 IP=IP-1
      KP=KP+1
      ZETA=0.0
      KWICH=2
      CALL RECIP(EA,FA,GA,WA,ZA,KP)
      GO TO 85
   77 DO 75 K=1,IP
      ZETA=PMIN(K)
      KP=KP+1
      CALL LINBA
   75 CONTINUE
      IF(FOPT(1).LT.0.5E-08) GO TO 85
      KP=KP+1
      ZETA=0.0
      KWICH=2
      CALL RECIP(EA,FA,GA,WA,ZA,KP)
      GO TO 85
   67 IF(FOPT(1).LT.0.5E-08) GO TO 85
      KP=KP+1
      ZETA=0.0
      RP=1.0
      KWICH=2
      CALL RECIP(EA,FA,GA,WA,ZA,KP)
   85 NPBR=IP*4+(KP-IP)*2
      NSBR=IS*4+(KS-IS)*2
      NPOL=NSBR+NPBR
      IF((NL-1)-NPOL)14,82,205
  205 IF(NTR.GE.1) GO TO 14
      GO TO(210,220,215),KWICH
  210 IF(NWI.GE.1) GO TO 240
      NWI=1
      GO TO 340
  240 IF(RR.GT.0) GO TO 245
      XW=X(KS-1)
      GO TO 350
  245 XW=X(KS)
      GO TO 350
  220 IF(NITCH.EQ.0) GO TO 225
      XW=X(KS)
      RR=WA(KP)
      GO TO 350
  225 IF(NWR.GE.1) GO TO 280
      NWR=1
      GO TO 340
  280 IF(RR.GT.0.0) GO TO 285
      XW=WA(KP)
      GO TO 350
```

```
285 XW=WA(KP-1)
    GO TO 350
215 IF(NWT.GE.1) GO TO 260
    NWT=1
    GO TO 340
260 RTPO(3)=1.0
    RTPO(2)=-(RR+1/RR)
    RTPO(1)=1.0
    NTPO=3
    GO TO 360
340 RR=-0.7
    IF(NWT.EQ.1) GO TO 341
    KSP=0
    CALL NEWT(ASUB,FSUB,NSUB)
341 KSP=1
    CALL NEWT(A,F,N)
    IF(RR.EQ.0.0) GO TO 14
    GO TO 85
350 RTPO(5)=1.0
    RTPO(4)=-(XW+1/XW+RR+1/RR)
    RTPO(3)=2.+(XW+1/XW)*(RR+1/RR)
    RTPO(2)=RTPO(4)
    RTPO(1)=1.0
    NTPO=5
360 ROPO(1)=F
    DO 295 I=1,N
295 ROPO(I+1)=A(I)
    NZZZ = N + 1
    CALL PDIV(REPO,NEPO,ROPO,NZZZ,RTPO,NTPO)
    PRINT2,(ROPO(I),I=1,4)
    FSPL=REPO(1)
    NEPOM1=NEPO - 1
    DO 300 I = 1, NEPOM1
300 ASPL(I)=REPO(I+1)
    NSPL=NEPO-1
    KSP=0
    RR=-0.7
    CALL NEWT(ASPL,FSPL,NSPL)
    KSP=1
    CALL NEWT(A,F,N)
    IF(RR.EQ.0.0) GO TO 14
    NTR=1
    GO TO 85
 82 PRINT21,NSBR
    PRINT22,NPBR
    PRINT5
    DO 81 I=1,KS
    IF(XC(I).EQ.0.0) GO TO 84
    SFAC(I)=X(I)**2+Y(I)**2
    PRINT6,X(I),Y(I),XC(I),YC(I)
 81 CONTINUE
    GO TO 93
 84 JI=I
```

```
      DO 86 I=JI,KS
   86 SFAC(I)=X(I)
      PRINT6,(X(I),Y(I),I=JI,KS)
      GO TO 93
   93 IF(KP.EQ.0) GO TO 92
      DO 91 I=1,KP
      IF(WAC(I).EQ.0.0) GO TO 94
      PFAC(I)=WA(I)**2+ZA(I)**2
      PRINT6,WA(I),ZA(I),WAC(I),ZAC(I)
   91 CONTINUE
      GO TO 92
   94 JI=I
      DO 96 I=JI,KP
   96 PFAC(I)=WA(I)
      PRINT6,(WA(I),ZA(I),I=JI,KP)
   92 CALL PRMUL(EB,E,EC,KS)
      DO 90 I=1,IN
   90 AAA(I)=XX(I)
      IA=IN
      DO 400 I=1,IA
      JJ=IA+1-I
      IF(AAA(JJ).NE.0.0) GO TO 410
  400 CONTINUE
  410 JA=IA-JJ
      IA=JJ
      IF(JA.EQ.0) GO TO 145
      DO 415 I=1,JA
  415 EC(KS+1-I)=1/EC(KS+1-I)
      CALL PRMUL(EB,E,EC,KS)
      DO 149 I=1,IN
  149 RAA(I)=XX(I)
      IR=IA
      GO TO 160
  145 DO 150 I=1,IA
  150 RAA(I)=AAA(I)
      IR=IA
  160 IOZ=IA+IR-1
      CALL PMPTY(ZOZ,IOZ,AAA,IA,RAA,IR)
      IF(NPBR.EQ.0) GO TO 165
      CALL PRMUL(EA,FA,GA,KP)
      DO 105 I=1,IN
  105 BB(I)=XX(I)
      IB=IN
      DO 430 I=1,IB
      JB=IB+1-I
      IF(BB(JB).NE.0.0) GO TO 110
  430 CONTINUE
  110 IB=JB
      DO 115 I=1,KP
      IF(EA(I).EQ.0) GO TO 120
      RFA(I)=FA(I)/GA(I)
  115 RGA(I)=1/GA(I)
      GO TO 125
```

```
120 JRB=I
    DO 435 I=JRB,KP
    RFA(I)=1.0
435 RGA(I)=1/GA(I)
125 CALL PRMUL(EA,RFA,RGA,KP)
    DO 130 I=1,IN
130 RBB(I)=XX(I)
    IRB=IB
140 IOP=IB+IRB-1
    CALL PMPTY(POR,IOP,BB,IB,RBB,IRB)
    ITT=IOZ+IOP-1
    CALL PMPTY(AAR,ITT,ZOZ,IOZ,POR,IOP)
    ISR=IA+IB-1
    CALL PMPTY(PSR,ISR,AAA,IA,BB,IB)
    GO TO 180
165 DO 170 I=1,IA
170 PSR(I)=AAA(I)
    ISR=IA
    DO 175 I=1,IOZ
    ITT=IOZ
175 AAR(I)=ZOZ(I)
180 DO 185 I=1,ITT
185 AAR(I)=F*AAR(I)
    PRINT10,ITT,(AAR(I),I=1,ITT)
    SBFAC=1.0
    DO 181 I=1,KS
181 SBFAC=SBFAC*SFAC(I)
    IF(KP.NE.0) GO TO 184
    PSFAC=DABS(SBFAC)
    GO TO 183
184 PBFAC=1.0
    DO 182 I=1,KP
182 PBFAC=PBFAC*PFAC(I)
    PSFAC = DABS(SBFAC*PBFAC)
183 DPMX =DSQRT(1+RHO+R1)
    DPMN =DSQRT(1+RHO-R1)
    PAPX=2/(DPMX+DPMN)
    SRF = PAPX * DSQRT(DABS(F)/PSFAC)
    DO 190 I=1,ISR
190 PSR(I)=SRF*PSR(I)
    SPD=DPMX*PAPX-1.0
    SSD ='PAPX*DSQRT(2*RHO)
    PRINT13,SPD,SSD
    PRINT4,(PSR(I),I=1,ISR)
    AAL(1)=F
    NLP=N+1
    DO 195 I=1,N
195 AAL(I+1)=A(I)
    PRINT 19
    CALL SFFT(PSR,ISR,NPOINT)
  5 FORMAT(//5X,'ROOTR',20X,'ROOTI')
  6 FORMAT((5X,E20.12,5X,E20.12))
 10 FORMAT(//I3/(5X,E20.12))
```

```
 13 FORMAT(//5X,'NEW PASSBAND DEVIATION=',F14.11
   1 /5X,'NEW STOPBAND DEVIATION=',F14.11)
 19 FORMAT(//5X,'MAGNITUDE RESPONSE')
 21 FORMAT(//5X,'NUMBER OF ROOTS IN STOPBAND=',I3)
 22 FORMAT(//5X,'NUMBER OF ROOTS IN PASSBAND=',I3)
    GO TO 9
 14 PRINT 8
  9 STOP
    END
    SUBROUTINE MILBA
    IMPLICIT REAL*8 (A-H,O-Z)
    COMMON/LI/A(128),F,N
    COMMON/MB/ZETA,ETA,RP,LL
    COMMON/BA/EB(100),E(100),EC(100),X(100),Y(100),XC(100),
   $ YC(100),KS
    DIMENSION B(128),C(128)
    REAL H,L,M
    DO 210 I = 1,128
    B(I) = 0.0
    C(I) = 0.0
210 CONTINUE
    T = -4.0 *DCOS(ETA*ZETA)
    J=0
    B(N)=A(N)
    NM4= N-4
    DO 5 I=1,NM4
  5 B(N-I)=A(N-I)-T*B(N-I+1)-(2+T**2/4)*B(N-I+2)-T*B(N-I+3)
   $ -B(N-I+4)
    C(N)=0
    DO 10 I=1,NM4
 10 C(N-I)=-B(N-I+1)-T*C(N-I+1)-T*B(N-I+2)/2-(2+T**2/4)*C(N-I+2)
   $ -B(N-I+3)-T*C(N-I+3)-C(N-I+4)
    V=A(3)-T*B(4)-(2+T**2/4)*B(5)-T*B(6)-B(7)
    U=A(2)-(2+T**2/4)*B(4)-T*B(5)-B(6)
    R=A(1)-T*B(4)-B(5)
    S=F-B(4)
    TT=T
    PC=V**2+U**2+R**2+S**2
 15 OPC=PC
    LP=0
 20 T=TT
    G=-B(4)-T*C(4)-T*B(5)/2-(2+T**2/4)*C(5)-B(6)-T*C(6)-C(7)
    H=-T*B(4)/2-(2+T**2/4)*C(4)-B(5)-T*C(5)-C(6)
    L=-B(4)-T*C(4)-C(5)
    M=-C(4)
    DELT=-(G*V+H*U+L*R+M*S)/(G**2+H**2+L**2+M**2)
    J=J+1
    IF(DABS(DELT).LT.0.5E-10) GO TO 30
    IF(J.GT.LL) GO TO 30
    LC=0
    ALPHA=1.0
 25 TT=T+ALPHA*DELT
    B(N)=A(N)
```

```
      DO 105 I=1,NM4
  105 B(N-I)=A(N-I)-TT*B(N-I+1)-(2+TT**2/4)*B(N-I+2)
     $ -TT*B(N-I+3)-B(N-I+4)
      C(N)=0
      DO 110 I=1,NM4
  110 C(N-I)=-B(N-I+1)-TT*C(N-I+1)-TT*B(N-I+2)/2
     $ -(2+TT**2/4)*C(N-I+2)-B(N-I+3)-TT*C(N-I+3)-C(N-I+4)
      V=A(3)-TT*B(4)-(2+TT**2/4)*B(5)-TT*B(6)-B(7)
      U=A(2)-(2+TT**2/4)*B(4)-TT*B(5)-B(6)
      R=A(1)-TT*B(4)-B(5)
      S=F-B(4)
      PC=V**2+U**2+R**2+S**2
      IF(PC.LT.OPC) GO TO 15
      LC=LC+1
      ALPHA=ALPHA/2
      IF(LC.LT.5) GO TO 25
      LP=LP+1
      IF(LP.LT.4) GO TO 20
   30 FZETA=(1/ETA)*DARCOS(-T/4)
      AR=T/2
      RTR=-AR/2
      TE=1-AR**2/4
      IF(TE.GE.0) GO TO 40
      IF(TE.LT.-0.5E-14) GO TO 35
      RTI=0
      GO TO 45
   35 TE=-TE
      X1 = RTR + DSQRT(TE)
      X2=RTR-DSQRT(TE)
      PRINT7,ZETA,FZETA,X1,X2
      GO TO 100
   40 RTI= DSQRT(TE)
   45 E(KS)=AR
      EB(KS)=1.0
      EC(KS)=1.0
      X(KS)=RTR
      Y(KS)=RTI
      XC(KS)=RTR
      YC(KS)=-RTI
      PRINT2,J
      PRINT3,DELT
      PRINT4,V,U,R,S
      PRINT6,ZETA,FZETA
    2 FORMAT(/5X,'NO. OF ITERATIONS=',I3)
    3 FORMAT(/5X,'CHANGE IN POLYNOMIAL COEFFICIENTS=',E20.12)
    4 FORMAT(/5X,'THE REMAINDERS'/5X,'V=',E20.12/5X,'U=',E20.12/5X,
     $ 'R=',E20.12/5X,'S=',E20.12)
    6 FORMAT(/5X,'INITIAL ZETA=',F10.7/5X,'FINAL ZETA=',F10.7)
    7 FORMAT(/5X,'INITIAL ZETA=',F10.7/5X,'FINAL ZETA=',F10.7//10X,
     $ 'REAL ROOTS'/5X,'X1=',E20.12/5X,'X2=',E20.12)
  100 RETURN
      END
      SUBROUTINE LINBA
```

```
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/LI/A(128),F,N
      COMMON/MB/ZETA,ETA,RP,LL
      COMMON/CD/EA(100),FA(100),GA(100),WA(100),ZA(100),WAC(100),
     $ ZAC(100),KP
      DIMENSION B(128),C(128),D(128)
      DO 210 I = 1,128
      B(I) = 0.0
      C(I) = 0.0
      D(I) = 0.0
  210 CONTINUE
      LTV = 0
      LRG=0
      P=RP
      LM=0
   35 Q=-2*DCOS(ETA*ZETA)
   60 J=0
      B(N)=A(N)
      NM2 = N-2
      DO 10 I=1,NM2
   10 B(N-I)=A(N-I)-P*Q*B(N-I+1)-(P**2)*B(N-I+2)
      C(N)=0
      DO 15 I=1,NM2
   15 C(N-I)=-Q*B(N-I+1)-P*Q*C(N-I+1)-2*P*B(N-I+2)-(P**2)*C(N-I+2)
      D(N)=0
      DO 20 I=1,NM2
   20 D(N-I)=-P*B(N-I+1)-P*Q*D(N-I+1)-(P**2)*D(N-I+2)
      R=A(1)-P*Q*B(2)-(P**2)*B(3)
      S=F-(P**2)*B(2)
      T=-Q*B(2)-P*Q*C(2)-2*P*B(3)-(P**2)*C(3)
      U=-P*B(2)-P*Q*D(2)-(P**2)*D(3)
      V=-2*P*B(2)-(P**2)*C(2)
      W=-(P**2)*D(2)
      TV=T*W-U*V
      IF(TV.NE.0) GO TO 130
      P=P-.005
      LTV=LTV+1
      IF(LTV.GT.10) GO TO 130
      GO TO 60
  130 DELP=(-R*W+S*U)/(T*W-U*V)
      IF(DABS(DELP).LT.1) GO TO 135
      P=P-.005
      LTV=LTV+1
      IF(LTV.GT.20) GO TO 135
      GO TO 60
  135 PT=P
      QT=Q
      PC=R**2+S**2
   25 OPC=PC
      LP=0
   30 P=PT
      Q=QT
      T=-Q*B(2)-P*Q*C(2)-2*P*B(3)-(P**2)*C(3)
```

```
      U=-P*B(2)-P*Q*D(2)-(P**2)*D(3)
      V=-2*P*B(2)-(P**2)*C(2)
      W=-(P**2)*D(2)
      DELP=(-R*W+S*U)/(T*W-U*V)
      DELQ=(-T*S+R*V)/(T*W-U*V)
      J=J+1
      IF(DABS(DELP).LT.0.5E-08.AND.DABS(DELQ).LT.0.5E-08) GO TO 55
      IF(J.GT.LL) GO TO 55
      L=0
      ALPHA=1.0
   40 PT=P+ALPHA*DELP
      QT=Q+ALPHA*DELQ
      IF(PT.LE.0) GO TO 45
      B(N)=A(N)
      DO 110 I=1,NM2
  110 B(N-I)=A(N-I)-PT*QT*B(N-I+1)-(PT**2)*B(N-I+2)
      C(N)=0
      DO 115 I=1,NM2
  115 C(N-I)=-QT*B(N-I+1)-PT*QT*C(N-I+1)-2*PT*B(N-I+2)
     $ -(PT**2)*C(N-I+2)
      D(N)=0
      DO 120 I=1,NM2
  120 D(N-I)=-PT*B(N-I+1)-PT*QT*D(N-I+1)-(PT**2)*D(N-I+2)
      R=A(1)-PT*QT*B(2)-(PT**2)*B(3)
      S=F-(PT**2)*B(2)
      PC=R**2+S**2
      IF(PC.LT.OPC) GO TO 25
   45 L=L+1
      ALPHA=ALPHA/2
      IF(L.LT.5) GO TO 40
      IF(PT.GT.0) GO TO 90
      LM=LM+1
      IF(LM.EQ.1) GO TO 50
      PRINT1,PT,P,DELP,QT,Q,DELQ
      GO TO 100
   50 LRG=LRG+1
      IF(LRG.NE.1) GO TO 51
      IF(RP.EQ.1) GO TO 51
      P=1.0
      LTV=0
      GO TO 35
   51 P=0.9
      LTV=0
      GO TO 35
   90 LP=LP+1
      IF(LP.LT.4) GO TO 30
   55 IF(P.LE.1.0) GO TO 65
      P=1/P
      LM=0
      GO TO 60
   65 FZETA = (1/ETA)*DARCOS(-Q/2)
      RP=P
      AR=Q*P
```

```
      AS=P**2
      RTR=-AR/2
      TE=AS-AR**2/4
      IF(TE.GE.0) GO TO 75
      IF(TE.LT.-0.5E-18) GO TO 70
      RTI=0
      GO TO 80
   70 TE=-TE
      X1=RTR+DSQRT(TE)
      X2=RTR-DSQRT(TE)
      PRINT3,ZETA,FZETA,X1,X2
      GO TO 100
   75 RTI=DSQRT(TE)
   80 FA(KP)=AR
      GA(KP)=AS
      EA(KP)=1.0
      WA(KP)=RTR
      ZA(KP)=RTI
      WAC(KP)=RTR
      ZAC(KP)=-RTI
      PRINT4,J
      PRINT5,DELP,DELQ
      PRINT6,R,S
      PRINT7,ZETA,FZETA
      PRINT8,RP
    1 FORMAT(/5X,(E20.12))
    3 FORMAT(/5X,'INITIAL ZETA=',F10.7/5X,'FINAL ZETA=',F10.7//10X,
     $ 'REAL ROOTS'/5X,'X1=',E20.12/5X,'X2=',E20.12)
    4 FORMAT(/5X,'NO. OF ITERATIONS=',I3)
    5 FORMAT(/5X,'CHANGE IN COEFFICIENTS'/5X,'DELP=',E20.12/5X,
     $ 'DELQ=',E20.12)
    6 FORMAT(/5X,'THE REMAINDERS'/5X,'R=',E20.12/5X,'S=',E20.12)
    7 FORMAT(/5X,'INITIAL ZETA=',F10.7/5X,'FINAL ZETA=',F10.7)
    8 FORMAT(/5X,'RADIUS=',E20.12)
  100 RETURN
      END
      SUBROUTINE PRMUL(EE,EF,EG,KK)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/PR/XX(128),IN
      DIMENSION EE(100),EF(100),EG(100),ZZ(128),YY(3)
      IN=3
      XX(1)=EG(1)
      XX(2)=EF(1)
      XX(3)=EE(1)
      IF(KK.EQ.1) GO TO 100
      DO 10 I=2,KK
      IK=3
      YY(1)=EG(I)
      YY(2)=EF(I)
      YY(3)=EE(I)
      IDIMZ=IN+IK-1
      DO 20 IZ=1,IDIMZ
   20 ZZ(IZ)=0
```

```
      DO 30 JJ=1,IK
      DO 30 II=1,IN
      NK=JJ+II-1
   30 ZZ(NK)=XX(II)*YY(JJ)+ZZ(NK)
      IN=IDIMZ
      DO 40 J=1,IN
   40 XX(J)=ZZ(J)
   10 CONTINUE
  100 RETURN
      END
      SUBROUTINE PMPTY(Z,IDMZ,X,IDMX,Y,IDMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Z(100),X(128),Y(128)
      IDMZ=IDMX+IDMY-1
      DO 10 I=1,IDMZ
   10 Z(I)=0
      DO 20 I=1,IDMX
      DO 20 J=1,IDMY
      K=I+J-1
   20 Z(K)=X(I)*Y(J)+Z(K)
      RETURN
      END
      SUBROUTINE FFT(C,D,N,NEWI,NSTUE,IK)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION C(512),D(512)
      DIMENSION CO(512),SI(512)
      INTEGER P
      PI2 = 8.0D0*DATAN(1.0D0)
      DO 13 J=1,9
      M=J
      IF(N-2**M)13,12,13
   13 CONTINUE
   12 CONTINUE
      IF(IK.NE.1) GO TO 433
      WRITE(6,434)
  434 FORMAT(1H0,'*****  INVERSE TRANSFORM  *****')
  433 CONTINUE
      NN=1
      NA=N/2
      XN=N
      NB=NA
      DO 16 I=1,N
      IA=0
      IC=I-1
      DO 15 J=1,M
      P=M-J
      P=2**P
      IB=IC/P
      IC=IC-IB*P
      P=2**(J-1)
   15 IA=IA+IB*P
      IA=IA+1
      IF(I.GE.IA) GO TO 166
```

```
      A=C(I)
      B=D(I)
      C(I)=C(IA)
      C(IA)=A
      D(I)=D(IA)
      D(IA)=B
  166 CONTINUE
   16 CONTINUE
      X=N
      X=PI2/X
      CO(1)=1.0
      SI(1)=0.0
      DO 3 J=1,M
      IF(NN.EQ.1)GO TO 5
      DO 6 L=1,NC
      I=NC-L+1
      K=2*I-1
      CO(K)=CO(I)
    6 SI(K)=SI(I)
      P=0
      DO 1 L=1,NC
      P=P+NB
       U = DCOS(X*P)
       V = -DSIN(X*P)
      IF(IK.EQ.1)V=-V
      IA=2*L
      CO(IA)=U
      SI(IA)=V
    1 P=P+NB
    5 P=1
      II=0
      DO 2 I=1,NA
      II=II+1
      KKK=P+NN
      A=C(KKK)*CO(II)-D(KKK)*SI(II)
      B=C(KKK)*SI(II)+D(KKK)*CO(II)
      C(KKK)=C(P)-A
      D(KKK)=D(P)-B
      C(P)=C(P)+A
      D(P)=D(P)+B
      IF(II.LT.NN)GO TO 77
      II=0
      P=P+NN
   77 P=P+1
    2 CONTINUE
      NC=NN
      NB=NB/2
      NN=2*NN
    3 CONTINUE
      RETURN
      END
      SUBROUTINE SFFT(X,N,NPOINT)
      IMPLICIT REAL*8 (A-H,O-Z)
```

```
      DIMENSION X(128),A(512),B(512)
      DO 10 I=1,N
      A(I)=X(I)
   10 B(I)=0
      MPOINT=NPOINT
      IF(MPOINT.GT.4) MPOINT=NPOINT/2+2
      IF(N.EQ.NPOINT) GO TO 13
      NP1 = N+1
      DO 28 I = NP1,NPOINT
      A(I)=0
   28 B(I)=0
   13 CALL FFT(A,B,NPOINT,NPOINT,NPOINT,-1)
      PRINT 14
      RNORM =DATAN2(B(2),A(2))
      PHASE1= RNORM
      DO 15 I2=1,MPOINT
      RMAG=DSQRT(A(I2)*A(I2)+B(I2)*B(I2))
      PHASE2 = DATAN2(B(I2),A(I2))
      DELAY = (PHASE2 - PHASE1)/RNORM
      PHASE1 = PHASE2
      PRINT 16,I2,RMAG,PHASE1,DELAY
   15 CONTINUE
   14 FORMAT(5X,'     MAGNITUDE       PHASE       GROUP PHASE DELAY')
   16 FORMAT (1X,I3,5X,F15.12,3X,F15.12,3X,F15.12)
      RETURN
      END
      SUBROUTINE RECIP(EK,FK,GK,XR,XI,KK)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/LI/A(128),F,N
      COMMON/MB/ZETA,ETA,RP,LL
      COMMON/QB/ASUB(128),FSUB,NSUB
      DIMENSION B(128),C(128),EK(100),FK(100),GK(100),XR(100),XI(100)
      DO 210 I = 1,128
      B(I) = 0.0
      C(I) = 0.0
  210 CONTINUE
      LTV = 0
      LDP = 0
      LCT = 0
      LP = 0
      D=-DCOS(ETA*ZETA)
      ND=D+D/2
      P=RP
      IF(P.LE.0.1) P=1.0
      LM=0
   35 J=0
      B(N)=A(N)
      NM2 = N-2
      DO 10 I=1,NM2
   10 B(N-I)=A(N-I)-ND*(P+1/P)*B(N-I+1)-B(N-I+2)
      C(N)=0
      DO 15 I=1,NM2
   15 C(N-I)=-ND*(1-1/(P**2))*B(N-I+1)-ND*(P+1/P)*C(N-I+1)
```

```
     $ -C(N-I+2)
      R=A(1)-ND*(P+1/P)*B(2)-B(3)
      S=F-B(2)
      T=-ND*(1-1/(P**2))*B(2)-ND*(P+1/P)*C(2)-C(3)
      V=-C(2)
      TV=T**2+V**2
      IF(TV.GT.0) GO TO 130
      P=P-.005
      LTV=LTV+1
      IF(LTV.GT.5) GO TO 130
      GO TO 35
  130 IF(LM.EQ.1) GO TO 135
      IF(LCT.EQ.2) GO TO 135
      DELP=-(T*R+V*S)/(T**2+V**2)
      IF(DABS(DELP).LT.0.25) GO TO 135
      LDP=LDP+1
      IF(LDP.GT.4) GO TO 140
      P=P-.005
      GO TO 33
  140 P=P-0.1
   33 IF(P.GE.0.1) GO TO 35
      LCT=LCT+1
      P=0.995
      LTV=0
      GO TO 35
  135 PT=P
      PC=R**2+S**2
   25 OPC=PC
      LP=0
   30 P=PT
      DELP=-(T*R+V*S)/(T**2+V**2)
      J=J+1
      IF(DABS(DELP).LT.0.5E-08) GO TO 55
      IF(J.GT.LL) GO TO 55
      L=0
      ALPHA=1.0
   40 PT=P+ALPHA*DELP
      IF(PT.LE.0) GO TO 45
      B(N)=A(N)
      DO 110 I=1,NM2
  110 B(N-I)=A(N-I)-ND*(PT+1/PT)*B(N-I+1)-B(N-I+2)
      C(N)=0
      DO 115 I=1,NM2
  115 C(N-I)=-ND*(1-1/(PT**2))*B(N-I+1)-ND*(PT+1/PT)*C(N-I+1)
     $ -C(N-I+2)
      R=A(1)-ND*(PT+1/PT)*B(2)-B(3)
      S=F-B(2)
      PC=R**2+S**2
      T=-ND*(1-1/(PT**2))*B(2)-ND*(PT+1/PT)*C(2)-C(3)
      V=-C(2)
      TVT=T**2+V**2
      IF(TVT.EQ.0) GO TO 45
      IF(PC.LT.OPC) GO TO 25
```

```
 45 L=L+1
    ALPHA=ALPHA/2
    IF(L.LT.5) GO TO 40
    IF(PT.GT.0) GO TO 90
    LM=LM+1
    IF(LM.EQ.1) GO TO 50
    GO TO 100
 50 P=0.5
    LTV=0
    GO TO 35
 90 LP=LP+1
    IF(LP.LT.4) GO TO 30
 55 IF(PC.GT.0.5E-6) GO TO 100
    IF(P.LE.1.0) GO TO 65
    P=1/P
 65 AS=-ND*P
    RP = P
    PP=1/P
    EK(KK)=0
    FK(KK)=1.0
    GK(KK)=-AS
    XR(KK)=AS
    XI(KK)=0
    NSUB=N-2
    FSUB=B(2)
    DO 66 I=1,NM2
 66 ASUB(I)=B(I+2)
    PRINT4,J
    PRINT5,DELP
    PRINT6,R,S
    PRINT7,RP
    GO TO 101
100 KK=KK-1
    PRINT8,PC
  4 FORMAT(/5X,'NO. OF ITERATIONS=',I3)
  5 FORMAT(/5X,'CHANGE IN ROOT=',E20.12)
  6 FORMAT(/5X,'THE REMAINDERS'/5X,'R=',E20.12/5X,'S=',E20.12)
  7 FORMAT(/5X,'RADIUS=',E20.12)
  8 FORMAT(/5X,'PC=',E20.12)
101 RETURN
    END
    SUBROUTINE NEWT(A,F,N)
    IMPLICIT REAL*8 (A-H,O-Z)
    COMMON/FI/RR,KSP
    COMMON/BA/EB(100),E(100),EC(100),X(100),Y(100),XC(100),
   $ YC(100),KS
    COMMON/CD/EA(100),FA(100),GA(100),WA(100),ZA(100),WAC(100),
   $ ZAC(100),KP
    COMMON/MB/ZETA,ETA,RP,LL
    DIMENSION A(128),FP(128),FDA(128)
    L = 0
    P=RR
    LM=0
```

```
35 J=0
   DO 5 I=1,N
 5 FP(I)=((P)**I)*A(I)
   FOA=F
   DO 10 I=1,N
10 FOA=FOA+FP(I)
   NM1 = N - 1
   DO 15 I=1,NM1
15 FDA(I)=(I+1)*A(I+1)*((P)**I)
   FDOA=A(1)
   DO 20 I=1,NM1
20 FDOA=FDOA+FDA(I)
   IF(FDOA.NE.0) GO TO 24
   P=P+.05
   GO TO 35
24 PT=P
   PC=FOA**2
25 OPC=PC
   LP=0
30 P=PT
   DELP=-FOA/FDOA
   J=J+1
   IF(DABS(DELP).LT.0.5E-12) GO TO 55
   IF(J.GT.LL) GO TO 55
   L=0
   ALPHA=1.0
40 PT=P+ALPHA*DELP
   DO 70 I=1,N
70 FP(I)=((PT)**I)*A(I)
   FOA=F
   DO 75 I=1,N
75 FOA=FOA+FP(I)
   DO 80 I=1,NM1
80 FDA(I)=(I+1)*A(I+1)*((PT)**I)
   FDOA=A(1)
   DO 85 I=1,NM1
85 FDOA=FDOA+FDA(I)
   IF(FDOA.EQ.0) GO TO 45
   PC=FOA**2
   IF(PC.LT.OPC) GO TO 25
45 L=L+1
   ALPHA=ALPHA/2
   IF(L.LT.5) GO TO 40
90 LP=LP+1
   IF(LP.LT.4) GO TO 30
55 IF(PC.LT.0.5E-15) GO TO 95
   LM=LM+1
   IF(LM.GT.3) GO TO 102
   RR=RR+0.5
   P=RR
   GO TO 35
102 IF(LM.GT.4) GO TO 101
   RR=-0.9
```

```
      P=RR
      GO TO 35
   95 IF(DABS(P).LE.1) GO TO 65
      P=1/P
      GO TO 35
   65 RR=P
      PRINT4,J
      PRINT6,DELP
      PRINT7,FOA
      PRINT8,RR
      PP=1/P
      IF(KSP.EQ.0) GO TO 101
      IF(P)105,101,115
  105 KS=KS+1
      EB(KS)=0.0
      E(KS)=1.0
      EC(KS)=-P
      X(KS)=P
      Y(KS)=0.0
      GO TO 101
  115 KP=KP+1
      EA(KP)=0.0
      FA(KP)=1.0
      GA(KP)=-P
      WA(KP)=P
      ZA(KP)=0.0
    4 FORMAT(/5X,'NO. OF ITERATIONS=',I3)
    6 FORMAT(/5X,'CHANGE IN APPROX. OF ROOT=',E20.12)
    7 FORMAT(/5X,'VALUE OF FUNCTION AT ROOT=',E20.12)
    8 FORMAT(/5X,'RADIUS=',E20.12)
  101 RETURN
      END
      SUBROUTINE REIM(EK,FK,GK,XR,XI,KK)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/LI/A(128),F,N
      COMMON/MB/ZETA,ETA,RP,LL
      COMMON/QB/ASUB(128),FSUB,NSUB
      DIMENSION D(128)
      DIMENSION B(128),C(128),EK(100),FK(100),GK(100),XR(100),XI(100)
      DO 210 I = 1,128
      B(I) = 0.0
      C(I) = 0.0
      D(I) = 0.0
  210 CONTINUE
      DD=-DCOS(ETA*ZETA)
      ND=DD+DD/2
      P=RP
      LM=0
   35 X=ND*(P+1/P)
      Y=1.0
      J=0
      B(N)=A(N)
      NM2 = N - 2
```

```
      DO 10 I=1,NM2
   10 B(N-I)=A(N-I)-X*B(N-I+1)-Y*B(N-I+2)
      C(N)=0
      DO 15 I=1,NM2
   15 C(N-I)=-B(N-I+1)-X*C(N-I+1)-Y*C(N-I+2)
      D(N)=0
      DO 20 I=1,NM2
   20 D(N-I)=-X*D(N-I+1)-B(N-I+2)-Y*D(N-I+2)
      R=A(1)-X*B(2)-Y*B(3)
      S=F-Y*B(2)
      T=-B(2)-X*C(2)-Y*C(3)
      U=-B(3)-X*D(2)-Y*D(3)
      V=-Y*C(2)
      W=-B(2)-Y*D(2)
      TV=T*W-U*V
      IF(TV.NE.0.0) GO TO 135
      P=P-.005
      LTV=LTV+1
      IF(LTV.GT.5) GO TO 135
      GO TO 35
  135 XT=X
      YT=Y
      PC=R**2+S**2
   25 OPC=PC
      LP=0
   30 X=XT
      Y=YT
      DELX=(-R*W+S*U)/(T*W-U*V)
      DELY=(-T*S+R*V)/(T*W-U*V)
      J=J+1
      IF(DABS(DELX).LT.0.5E-10.AND.DABS(DELY).LT.0.5E-10) GO TO 55
      IF(J.GT.LL) GO TO 55
      L=0
      ALPHA=1.0
   40 XT=X+ALPHA*DELX
      YT=Y+ALPHA*DELY
      IF(YT.LE.0) GO TO 45
      B(N)=A(N)
      DO 110 I=1,NM2
  110 B(N-I)=A(N-I)-XT*B(N-I+1)-YT*B(N-I+2)
      C(N)=0
      DO 115 I=1,NM2
  115 C(N-I)=-B(N-I+1)-XT*C(N-I+1)-YT*C(N-I+2)
     $ -(PT**2)*C(N-I+2)
      D(N)=0
      DO 120 I=1,NM2
  120 D(N-I)=-XT*D(N-I+1)-B(N-I+2)-YT*D(N-I+2)
      R=A(1)-XT*B(2)-YT*B(3)
      S=F-YT*B(2)
      PC=R**2+S**2
      T=-B(2)-XT*C(2)-YT*C(3)
      U=-B(3)-XT*D(2)-YT*D(3)
      V=-YT*C(2)
```

```
      W=-B(2)-YT*D(2)
      TVT=T*W-U*V
      IF(TVT.EQ.0.0) GO TO 45
      IF(PC.LT.OPC) GO TO 25
   45 L=L+1
      ALPHA=ALPHA/2
      IF(L.LT.5) GO TO 40
      IF(YT.GT.0) GO TO 90
      GO TO 100
   90 LP=LP+1
      IF(LP.LT.4) GO TO 30
   55 IF(PC.GE.0.5E-6) GO TO 100
      IF((ND*X).GT.0.0) GO TO 65
      PRINT 11
      GO TO 100
   65 TE=X**2/4-Y
      REP=-X/2
      IF(TE.GE.0.5E-20) GO TO 70
      IF(TE.LE.-0.5E-20) GO TO 75
      P=-X/2
      IF(DABS(P).LE.1.0) GO TO 76
      IF(DABS(P).LE.(1.0+0.5E-12)) GO TO 77
      PRINT7,P
      GO TO 100
   77 P=-ND*1.0
   76 RTR=P
      RTI=0.0
      GO TO 85
   70 AROT=REP-DSQRT(TE)
      IF(DABS(AROT).LE.1) GO TO 80
      AROT=REP+DSQRT(TE)
   80 RTR=AROT
      OTR=1/RTR
      RTI=0
      GO TO 85
   75 TE=-TE
      RTR=-X/2
      RTI=DSQRT(TE)
      CTI=-RTI
      PRINT 9
      GO TO 100
   85 RP=RTR
      EK(KK)=0.0
      FK(KK)=1.0
      GK(KK)=-RTR
      XR(KK)=RTR
      XI(KK)=0.0
      NSUB=N-2
      FSUB=B(2)
      DO 66 I=1,NM2
   66 ASUB(I)=B(I+2)
      PRINT4,J
      PRINT5,DELX,DELY
```

```
      PRINT6,R,S
      PRINT7,RP
      GO TO 101
  100 KK=KK-1
      PRINT8,PC
    4 FORMAT(/5X,'NO. OF ITERATIONS=',I3)
    5 FORMAT(/5X,'CHANGE IN COEFFICIENTS'/5X,'X COEFF=',E20.12/5X,
     $ 'CONST TERM=',E20.12)
    6 FORMAT(/5X,'THE REMAINDERS'/5X,'R=',E20.12/5X,'S=',E20.12)
    7 FORMAT(/5X,'RADIUS = ',E20.12)
    8 FORMAT(/5X,'PC=',E20.12)
    9 FORMAT(/5X,'COMPLEX ROOTS')
   11 FORMAT(/5X,'SIGN DISAGREES')
  101 RETURN
      END
      SUBROUTINE PDIV(P,IDIMP,X,IDIMX,Y,IDIMY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION P(128),X(128),Y(30)
      IDIMP=IDIMX-IDIMY+1
      IDIMX=IDIMY-1
      I=IDIMP
   70 II=I+IDIMX
      P(I)=X(II)/Y(IDIMY)
      DO 80 K=1,IDIMX
      J=K-1+I
      X(J)=X(J)-P(I)*Y(K)
   80 CONTINUE
      I=I-1
      IF(I)90,90,70
   90 RETURN
      END
```

APPENDIX B

COMPLEX APPROXIMATION FILTER COMPUTER PROGRAM

```
      IMPLICIT REAL*8(A-H,O-Z)
C     PROGRAM TO FIND BEST CHEBYSHEV COMPLEX APPROXIMATION TO A
C     DESIRED LOW PASS FILTER ON A FINITE POINT SET USING THE
C     METHOD OF LAWSON ( REFERENCE - J.R. RICE, THE APPROXIMATION
C     OF FUNCTIONS, VOL. II PAGES 298-304)
C
C     J. D. FISHER   RICE UNIVERSITY, HOUSTON, TEXAS   MAY, 1973
C
C    THE LAWSON ALGORITHM USES A SUCCESSION OF BEST WEIGHTED LEAST
C      SQUARES APPROXIMATION.
C      AT EACH STEP OF THE ALGORITHM THE OPTIMAL LEAST
C    SQUARES APPROXIMATION IS GIVEN BY
C           X = (A**H*Q*A)**-1*A**H*Q*D   WHERE
C     THE (I,K) ELEMENT OF A IS GIVEN BY EXP(-J*(K-1)*((I-1)*2 PI/M)
C                    K = 1....N WHERE N IS THE LENGTH OF THE FILTER
C                    I=1....M  WHERE M IS THE NUMBER OF POINTS IN THE
C                      FINITE POINT SET ONE IS APPROXIMATING ON
C
C     A**H DENOTES A CONJUGATED AND TRANSPOSED
C
C     Q IS A MXM WEIGHTING MATRIX
C     AND D IS THE DESIRED RESPONSE
C
C     THE INPUT IS CURRENTLY SET UP TO HANDLE THE INPUT
C      OF D IN TWO DIFFERENT MANNERS
C
C     IN THE FIRST CASE A DESIRED LOWPASS  FILTER IS SPECIFIED
C     WITH A GIVEN LINEAR SLOPE WHICH IS SOME FRACTION OF NORMAL
C     LINEAR PHASE SLOPE
C          THE CARD SET UP IS AS FOLLOWS
C     CARD 1 = NH = FILTER LENGTH (ENTERED FIRST 3 COLUMNS)
C     CARD 2 = NPOINT= = NUMBER OF POINTS ON (0...5) (FIRST 3 COLUMNS)
C          WHERE (0...5) DENOTES THE NORMALIZED FREQUENCY VARIABLE
C          BOTH 0. AND .5 ARE INCLUDED IN THE POINT SET
C     CARD 3 = FP = PASSBAND CUTOFF (F10.5)
C     CARD 4 = FS = STOPBAND CUTOFF (F10.5)
C     CARD 5 = NUMIT = NUMBER OF ITERATIONS BEFORE STOPPING(FIRST 3
C                COLUMNS)
C     CARD 6 = ID = INPUT OPTION = 0 FOR LINEAR SLOPE FACTOR
C     CARD 7 = SSLOPE = LINEAR SLOPE FACTOR(F14.10)
C
C     HENCE A SAMPLE CARD SET UP WOULD BE
C
C     009
C     065
C     .10
C     .20
C     050
C     0
C     .75
C
C     WHICH WOULD SPECIFY A LENGTH 9 FILTER ON A FINITE POINT SET
```

```
C      OF 65 EVENLY SPACED POINTS INCLUDING 0. AND .5 WITH A
C      PASSBAND CUTOFF OF .10 AND A STOPBAND CUTOFF OF 0.2
C      PROCEEDING FOR 50 ITERATIONS TRYING TO APPROXIMATE A
C      LOWPASS FILTER WHOSE SLOPE FACTOR IS 0.75 THAT OBTAINED BY AN
C      EXACTLY LINEAR PHASE FILTER
C
C      THE SECOND MODE OF INPUT ALLOWS ONE TO ENTER HIS OWN
C      DESIRED PHASE TO GO WITH A DESIRED MAGNITUDE OF
C      ONE IN THE PASSBAND
C      CARDS 1 - 5 ARE THE SAME AS BEFORE
C
C      CARD 6 = ID = OPTION CARD = 1
C
C      CARD 7 - ON = THE VALUE OF THE DESIRED  PHASE (F10.5)
C
C      HENCE A SAMPLE CARD INPUT  WOULD APPEAR
C
C      009
C      065
C      .1
C      .20
C      050
C      1
C
C      AND THEN THE CARDS WITH THE PHASE SPECIFICATIONS (IN RADIANS)
C      .1
C      .2
C      .3 ETC
C
C       THIS SET OF SPECIFICATIONS WOULD DENOTE A FILTER OF LENGTH
C      9 ON A POINT SET OF 65 POINTS. WITH PASS AND STOPBAND CUTOFFS OF
C      .1 AND .2 RESPECTIVELY, HAVING A DESIRED RESPONSE AT THE FIRST
C      POINT OF MAGNITUDE 1 PHASE .1 RADIANS. AT THE SECOND POINT
C      MAGNITUDE 1. PHASE .2. AT THE THIRD POINT MAGNITUDE 1
C      PHASE .3 ETC.
C
C           * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C      AT THIS TIME THE DIMENSION OF THE ARRAYS MUST BE CHANGED TO
C      REFLECT THE LENGTH OF THE IMPULSE RESPONSE CALLED FOR ---
C      IN THIS CASE THE MATRICES ARE SET UP TO HANDLE A FILTER OF LENGTH
C        9 ON A MAXIMUM DENSE GRID OF 129 POINTS ON THE HALF INTERVAL
C
      COMPLEX *16 A(129,9),ACONG(129,9),D(129),RNEWA(129,9)
      COMPLEX *16 R(129),TEMPR(129),SUM
       COMPLEX *16 COMPN,RHOLD1,RHOLD2
      REAL *8 WT(129), DEV(129),TWT(129),PWT(129),APASS(9,9),BPASS(9),
     1 TEMPH(9),TEMPD(129)
       COMMON/BACK1/REALN
       COMMON/BACK2/AIMAGN
       COMMON/DOWN/COMPN
      AAA = 0.0D0
      BBB = 1.D0
      CCC=2.D0
```

```
      WRITE (6,1)
    1 FORMAT ('  ENTER LENGTH OF IMPULSE (I3)')
      READ (5,102) NH
  102 FORMAT (I3)
      WRITE (6,2)
    2 FORMAT ('  ENTER NUMBER OF POINTS ON HALF INTERVAL(I3)')
      READ (5,102) NPOINT
      WRITE (6,3)
    3 FORMAT ('  ENTER PASSBAND CUTOFF FP (F10.5)')
       READ(5,354) FP
      WRITE (6,4)
    4 FORMAT ('  ENTER STOPBAND CUTOFF FS (F10.5)')
      READ(5,354) FS
      WRITE (6,5)
    5 FORMAT (' ENTER MAXIMUM NUMBER OF ITERATIONS (I3)')
      READ (5,102) NUMIT
        WRITE (6,350)
  350 FORMAT('  TO ENTER OWN DESIRED PHASE ENTER A 1')
        READ(5,106) ID
      IF (ID.EQ.1) GO TO 222
       WRITE(6,506)
  506  FORMAT('  ENTER SLOPE FACTOR(F10.5)')
       READ (5,180) SSLOPE
      SLOPE = SSLOPE
  106 FORMAT (I1)
C
  222 TCHEBY = 1.D5
      IQ=NPOINT
       RNTOTD=NPOINT
      RINT=.5/(RNTOTD-1.)
      START=0.0
      IGOT = 0
      IGOT2 = 0
      DO 31 I = 1,NPOINT
       IF(IGOT2.EQ.1) GO TO 31
      IF(FP.GE.START.OR.IGOT.EQ.1) GO TO 32
       IGOT=1
       NFL = I
      IF(DABS(FP-START+RINT).LT.0.1*(RINT)) NFL=I-1
      FRACT = DABS((NFL-2.)*RINT - FP)/RINT
   32  IF(FS.GE.START) GO TO 33
      IGOT2=1
      NFS = I-1
      IF(DABS(FS-NFS*RINT).LT.0.1*RINT) NFS=I
   33 START = START + RINT
   31 CONTINUE
      RINDEX = -1.D0
      IFIRST = 0
      ITER = 0
      NITER=0
```

```
      PI = 3.1415926535897900
      RMULT = PI * 2. * RINT
      M = NH
      MM = M*M
      N = 1
      EPS = 1.D-5
       NTRAN = NFS - NFL - 1
      N2 = NPOINT - (NFS - NFL)
       IQ=N2 + 1
       RIQ=IQ
       WTI=.5/RIQ
C
C   GENERATE BASIS FUNCTIONS  (MATRICES A AND ACONG)
C
C
       RNH = NH - 1
      DEG = SLOPE * (RNH/CCC) * RMULT
      DO 55555 J = 1, NPOINT
      WT(J) = WTI
      PWT(J) = WTI
55555 CONTINUE
      NFLM1 = NFL - 1
      DO 6 K = 1, NH
      RKK = K - 1
      DO 40 J = 2,NFLM1
      RJJ = J - 1
      OMEGA = RJJ*RMULT
      RAA =DCOS(RKK * OMEGA)
      RAAI =DSIN(RKK * OMEGA)
      A(J,K) =DCMPLX(RAA,-RAAI)
      ACONG(J,K) =DCMPLX(RAA,RAAI)
   40 CONTINUE
      IINFL = NFL + 2
      DO 41 J = IINFL,N2
       RJJ = J + NTRAN - 1
      OMEGA = RJJ*RMULT
      RAA =DCOS(RKK * OMEGA)
      RAAI =DSIN(RKK * OMEGA)
      A(J,K) =DCMPLX (RAA,-RAAI)
      ACONG(J,K) =DCMPLX(RAA,RAAI)
   41 CONTINUE
      A(1,K) = CMPLX(1.0,0.0)
      A(IQ           ,K) = DCMPLX(DCOS((K-1)*PI),AAA)
      ACONG(1,K) = A(1,K)
      ACONG(N2+1,K) = A(N2+1,K)
      RAA = DCOS(RKK*FP*2.*PI)
      RAAI=DSIN(RKK*FP*2.*PI)
      A(NFL,K) = DCMPLX(RAA,-RAAI)
      ACONG(NFL,K) = DCMPLX(RAA,RAAI)
      RAA = DCOS(RKK*FS*2.*PI)
      RAAI=DSIN(RKK*FS*2.*PI)
        A(NFL+1,K) =DCMPLX(RAA,-RAAI)
      ACONG(NFL+1,K) = DCMPLX(RAA,RAAI)
```

```
    6 CONTINUE
C
C  GENERATE CHARACTERISTIC TO BE APPROMIMATED  (D VECTOR)
        DO 2222 I = 1,IQ
2222  D(I) = DCMPLX (AAA,AAA)
       IF(ID.EQ.1) GO TO 351
 354      FORMAT(F10.5)
  507 DO 11111 I = 2,NFLM1
      RT = I - 1
      DR =DCOS(RT*DEG)
      DI =-DSIN(RT*DEG)
      D(I) =DCMPLX(DR, DI)
11111 CONTINUE
         D(1) = DCMPLX(BBB,AAA)
      RT = NFLM1 + FRACT - 1
      DR = DCOS(RT*DEG)
      DI = - DSIN(RT*DEG)
      D(NFL) = DCMPLX(DR,DI)
      GO TO 357
C
C NFL IS THE NUMBER OF POINTS IN THE PASSBAND
C   IQ IS THE TOTAL NUMBER OF POINTS LEFT TO DO THE
C      APPROXIMATION ON ONCE THE TRANSITION POINTS
C     HAVE BEEN TAKEN OUT OF THE POINT SET
C
C
C     TO MODIFY THE WAY THE PROGRAM HANDLES YOUR OWN ENTERED
C     DESIRED RESPONSE MODIFY THE NEXT 8 CARDS ACCORDINGLY
 351      RMAG=1.
          DO 353 I = 1,NFL
        READ(5,354) PHASE
        RHOLD1 = DCMPLX(RMAG,AAA)
        RHOLD2 = DCMPLX(AAA,PHASE)
             RHOLD1 = RHOLD1 * CDEXP(RHOLD2)
        D(I) = RHOLD1
 353       CONTINUE
357        CONTINUE
      WRITE(6,48)
   48 FORMAT('  THE FOLLOWING PRINT OUT IS THE MAX DEVIATION AFTER EACH
     1  ITERATION')
C
C  NOW HAVE ALL MATRICES, JUST MULTIPLY OUT AND DONE
C
  300 CHEBY = 0.0
      DO 13 I8= 1,NH
      DO 12 I7 = 1, IQ
   12 RNEWA(I7,I8) = ACONG(I7,I8) * WT(I7)
   13 CONTINUE
      DO 7 I1 = 1,NH
      DO 8 I2 = 1,NH
        SUM = DCMPLX(AAA,AAA)
      DO 9 I3 = 2, N2
    9 SUM = SUM + RNEWA(I3,I1) * A(I3,I2)
```

```
      COMPN = SUM
       CALL SORRY
       TEMP1 = REALN
       COMPN = RNEWA(1,I1) * A(1,I2)
       CALL SORRY
       TEMP2 = REALN
       COMPN = RNEWA(N2 + 1,I1) * A(N2+1,I2)
       CALL SORRY
       APASS(I1,I2) = 2.*TEMP1 + TEMP2 + REALN
    8 CONTINUE
    7 CONTINUE
      DO 10 I4 = 1,NH
        SUM = DCMPLX(AAA,AAA)
      DO 11 I5 = 2,NFL
   11 SUM = SUM + RNEWA(I5,I4) * D(I5)
        COMPN = SUM
        CALL SORRY
        TEMP1 = REALN
        COMPN = RNEWA(1,I4) * D(1)
       CALL SORRY
       TEMP2 = REALN
       BPASS(I4) = 2.*TEMP1 + TEMP2
   10 CONTINUE
C  BPASS HAS ANSWER APASS IS DESTROYED
      CALL GLEG (M, MM, N, EPS, IER, APASS, BPASS)
      IF(IER.EQ.-1) GO TO 9999
C
C  NOW CALCULATE WHAT HAVE AND COMPARE IT WITH WHAT WE WANTED
C
C RESPONSE = A*X
C
  215 DO 21  J1 = 1, IQ
        SUM = DCMPLX(AAA,AAA)
      DO 22 J2 = 1,NH
   22 SUM = SUM + A(J1,J2)  * BPASS(J2)
      R(J1) = SUM
   21 CONTINUE
      DO 99 J3 = 1,IQ
      DEV(J3) =CDABS(R(J3) - D(J3))
      TEMP = DEV(J3)
      IF ((CHEBY - TEMP).LT.0.0) CHEBY = TEMP
   99 CONTINUE
      WRITE (6,120) CHEBY
      IF (CHEBY.GE.TCHEBY) GO TO 997
      DO 996 III = 1,NH
  996 TEMPH(III) = BPASS(III)
      TCHEBY = CHEBY
      DO 995 II2= 1,IQ
      TEMPR(II2) = R(II2)
      TEMPD(II2)=DEV(II2)
  995 CONTINUE
  997 DO 316 L4 = 1,IQ
  316 IF(WT(L4).EQ.0.0) WT(L4)=PWT(L4)
```

```
      DO 302 L1 = 1,IQ
  302 TWT(L1) = WT(L1)*DEV(L1)
      RSUM = 0.0
      DO 303 L2 = 1,IQ
  303 RSUM=RSUM + TWT(L2)
      DO 304 L3 = 1,IQ
  304 WT(L3) = TWT(L3)/RSUM
  191 FORMAT (E10.4)
  410 DO 311  L5 = 1, IQ
  311 PWT(L5) = WT(L5)
  318 ITER= ITER + 1
      IF (ITER.EQ.NUMIT) GO TO 200
      GO TO 300
  200 DO 20 I = 1,NH
      WRITE(6,110) I,TEMPH(I)
   20 CONTINUE
      WRITE(6,140)
      DO 30 K1 = 1,IQ
       COMPN = TEMPR(K1)
       CALL SORRY
       RR = REALN
       RIM = AIMAGN
       RMAG = DSQRT(RR*RR + RIM*RIM)
      RPHASE =DATAN2(RIM,RR)
      RINDEX =RINDEX + 1
      FR = RINT * RINDEX
      IF(K1.EQ.NFL) RINDEX = RINDEX + NTRAN
      IF(K1.EQ.NFL) FR=FP
      IF(K1.EQ.NFL+1) FR=FS
      WRITE(6,201) K1,FR,RMAG,RPHASE
   30 CONTINUE
      WRITE(6,42) NH,IQ,FP,FS
      IF(ID.EQ.0) WRITE(6,47)  SLOPE
   47 FORMAT('  WITH A PHASE FACTOR OF  ',F10.5)
      WRITE(6,796)
  796 FORMAT('  THE ERROR VECTOR ON THESE POINTS IS')
      DO 45 I = 1, IQ
   45 WRITE(6,160) TEMPD(I)

      GO TO 10000
 9999 PRINT 188
  188 FORMAT (' ERROR IN SUBR. GLEG')
      GO TO 10000
  110 FORMAT('  H(',I3,') =',F14.8)
  120 FORMAT (' E = ', F10.8)
  140 FORMAT('                  MAGNITUDE    PHASE')
  160 FORMAT(F14.10)
  170 FORMAT(I5)
  180 FORMAT(F14.10)
  186 FORMAT (2F12.7)
  187 FORMAT (I2)
  201 FORMAT(I3,3F10.5)
   42 FORMAT(1X,' LENGTH = ',I3,' ON ',I4,' POINTS   WITH FP = ',F10.5
```

```
     1 ,' AND FS = ',F10.5)
   46 FORMAT(F10.5)
10000 DUMMNY = AAA
      STOP
      END
      SUBROUTINE GLEG (M, MM, N, EPS, IER, A, R)
      IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION R(5),A(25)
      IF(M)23,23,1
C
C     SEARCH FOR GREATEST ELEMENT IN MATRIX A
    1 IER=0
      PIV=0.
      MM=M*M
      NM=N*M
      DO 3 L=1,MM
      T1 =DABS(A(L))
      IF(T1-PIV) 3,3,2
    2 PIV = T1
      I=L
    3 CONTINUE
      TOL=EPS*PIV
C     A(I) IS PIVOT ELEMENT, PIV CONTAINS THE ABSOLUTE VALUE OF A(I).
C
C
C     START ELIMINATION LOOP
      LST=1
      DO 17 K=1,M
C     TEST ON SINGULARITY
C
      IF(PIV)23,23,4
    4 IF(IER)7,5,7
    5 IF(PIV-TOL)6,6,7
    6 IER=K-1
    7 PIVI=1./A(I)
      J=(I-1)/M
      I=I-J*M-K
      J=J+1-K
C     I+K IS ROW-INDEX, J+K COLUMN-INDEX OF PIVOT ELEMENT
C
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R
      DO 8 L=K,NM,M
      LL=L+I
      TB=PIVI*R(LL)
      R(LL)=R(L)
    8 R(L)=TB
C
C     IS ELIMINATION TERMINATED
      IF (K-M) 9, 18, 18
C
C     COLUMN INTERCHANGE IN MATRIX A
    9 LEND=LST+M-K
      IF(J)12,12,10
```

```
   10 II=J*M
      DO 11 L=LST, LEND
      TB=A(L)
      LL=L+II
      A(L)=A(LL)
   11 A(LL)=TB
C
C     ROW INTERCHANGE AND PIVOT ROW REDUCTION IN MATRIX A
   12 DO 13 L=LST,MM,M
      LL=L+I
      TB=PIVI*A(LL)
      A(LL)=A(L)
   13 A(L)=TB
C
C     SAVE COLUMN INTERCHANGE INFORMATION
      A(LST) = J
C
C     ELEMENT REDUCTION AND NEXT PIVOT SEARCH
      PIV=0.
      LST=LST+1
      J=0
      DO 166 II=LST,LEND
      PIVI=-A(II)
      IST=II+M
      J=J+1
      DO 15 L=IST,MM,M
      LL=L-J
      A(L)=A(L)+PIVI*A(LL)
      T1 =DABS(A(L))
      IF(T1-PIV) 15,15,14
   14 PIV = T1
      I=L
   15 CONTINUE
      DO 16 L=K,NM,M
      LL=L+J
   16 R(LL)=R(LL)+PIVI*R(L)
  166 CONTINUE
   17 LST=LST+M
C     END OF ELIMINATION LOOP
C
C
C     BACK SUBSTITUTION AND BACK INTERCHANGE
   18 IF(M-1)23,22,19
   19 IST=MM+M
      LST=M+1
      DO 21 I=2,M
      II=LST-I
      IST=IST-LST
      L=IST-M
      L = A(L) + .5
      DO 21 J=II,NM,M
      TB=R(J)
      LL=J
```

```
      DO 20 K=IST,MM,M
      LL=LL+1
   20 TB=TB-A(K)*R(LL)
      K=J+L
      R(J)=R(K)
   21 R(K)=TB
   22 RETURN
C
C
C     ERROR RETURN
   23   IER=-1
      RETURN
      END
        SUBROUTINE SORRY
        REAL *8 COMP(2),A,B
C     SUBROUTINE SORRY MERELY FINDS THE REAL AND IMAGINARY
C     PARTS OF COMPLEX * 16 NUMBERS
C
        COMMON/BACK1/A
        COMMON/BACK2/B
        COMMON/DOWN/COMP
        DO 1 I = 1,2
        A = COMP(1)
        B= COMP(2)
 1        CONTINUE
        RETURN
        END
```

CHAPTER VII

REFERENCES

[1] J. F. Kaiser, "Digital Filters," in System Analysis by Digital Computer, F. F. Kuo and J. F. Kaiser, Eds., New York: Wiley, 1966, pp. 218-285.

[2] L. R. Rabiner, B. Gold, and C. A. McGonegal, "An Approach to the approximation problem for nonrecursive digital filters," IEEE Transactions on Audio and Electroacoustics, vol. AU-18, June 1970, pp. 83-106.

[3] E. Hofstetter, A. V. Oppenheim, and J. Siegel, "A new technique for the design of nonrecursive digital filters," presented at the Fifth Annual Princeton Conference on Information Sciences and Systems, March 1971.

[4] T. W. Parks and J. H. McClellan, "Chebyshev Approximation for Non-Recursive Digital Filters with Linear Phase," IEEE Transactions on Circuit Theory, vol. CT-19, March 1972, pp. 189-194.

[5] F. W. Cheney, Introduction to Approximation Theory. New York: McGraw Hill, 1966, pp. 72-100.

[6] Bernard Gold and Charles M. Rader, Digital Processing of Signals, New York: McGraw Hill, 1969.

[7] O. Hermann and N. Schuessler, "Design of non-recursive digital filters with minimum phase," Electronic Letters, vol. 6, No. 11, pp. 329-330.

[8] G. G. Lorentz, Approximation of Functions. New York: Holt, Rinehart and Winston, 1966.

[9] J. R. Rice, The Approximations of Functions. Vol. II, Reading, Mass: Addison Wesley, 1969, pp. 298-304.

[10] R. W. Hamming, Numerical Methods for Scientists and Engineers. New York: McGraw Hill, 1962.

[11] O. Herrmann, L. R. Rabiner and D. S. K. Chan, "Practical design rules for optimum finite impulse response digital filters," to appear in Audio and Electro acoustics.

[12] T. W. Parks and J. H. McClellan, "A program for the design of linear phase finite impulse response digital filters," IEEE Transactions on Audio and Electroacoustics, Vol. AU-20, August 1972, pp. 195-199.